2006-03

A performance analysis of routing protocols for adhoc networks

Pore, Ghee Lye.

Monterey California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

A PERFORMANCE ANALYSIS OF ROUTING PROTOCOLS FOR AD-HOC NETWORKS

by

Ghee Lye Pore

March 2006

Thesis Advisor: John C. McEachen
Second Reader: Weilian Su

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2006	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE: A Performance Analysis of Routing Protocols for Ad-hoc Networks	**5. FUNDING NUMBERS**
6. AUTHOR(S) Ghee Lye Pore	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

A mobile ad-hoc network (MANET) is an autonomous system of mobile nodes connected by wireless links. The performance of MANET is related to the efficiency of the routing protocols in adapting to frequently changing network topology and link status. This thesis addresses the issue by comparing the relative performance of three key ad-hoc routing protocols: Destination-sequenced Distance Vector (DSDV), Ad-hoc On-demand Distance Vector (AODV) and Optimized Link State Routing (OLSR). The protocols are tested based on two scenarios, namely, tactical networks for ships and sensor-based network nodes. The objective is to validate the scalability and effectiveness of the protocols. Four performance metrics were measured by varying the maximum speed of mobile hosts, network size and traffic load, to assess the routing capability and protocol efficiency. The simulation results indicate that AODV performs better than OSLR and DSDV in the first scenario. Although OLSR also performed relatively well, the associated high routing overhead is the dominant reason for not choosing it. On the other hand, OLSR emerged as the protocol of choice for sensor networks, where the high routing overhead is counteracted by consistently better performance in all other metrics. Due to the slow evolution of the sensor network topology, OLSR performed satisfactorily for best effort traffic but needed subtle adjustments to balance between latency and bandwidth to meet the requirements of delay-sensitive applications. Lastly, default parameters of OLSR were tweaked and recommendations were made with results that showed promising ways to further improve the performance of OLSR in sensor networks, albeit not as significantly as in the tactical networks for the ship case.

14. SUBJECT TERMS Ad-hoc networks, Wireless Networks, Mobile Networks, Routing Protocols, Network Simulation, Performance Evaluation.	**15. NUMBER OF PAGES** 113
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

A PERFORMANCE ANALYSIS OF ROUTING PROTOCOLS FOR AD-HOC NETWORKS

Ghee Lye Pore
Civilian, Defence Science & Technology Agency (DSTA), Singapore
Bachelor of Engineering (Hons), University of Glasgow, United Kingdom, 1995

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 2006

Author: Ghee Lye Pore

Approved by: John C. McEachen
Thesis Advisor

Weilian Su
Second Reader

Jeffrey B. Knorr
Chairman, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A mobile ad-hoc network (MANET) is an autonomous system of mobile nodes connected by wireless links. The performance of MANET is related to the efficiency of the routing protocols in adapting to frequently changing network topology and link status. This thesis addresses the issue by comparing the relative performance of three key ad-hoc routing protocols: Destination-sequenced Distance Vector (DSDV), Ad-hoc On-demand Distance Vector (AODV) and Optimized Link State Routing (OLSR). The protocols were tested based on two scenarios, namely, tactical networks for ships and sensor-based network nodes. The objective is to validate the scalability and effectiveness of the protocols. Four performance metrics were measured by varying the maximum speed of mobile hosts, network size and traffic load, to assess the routing capability and protocol efficiency. The simulation results indicate that AODV performs better than OSLR and DSDV in the first scenario. Although OLSR also performed relatively well, the associated high routing overhead is the dominant reason for not choosing it. On the other hand, OLSR emerged as the protocol of choice for sensor networks, where the high routing overhead is counteracted by consistently better performance in all other metrics. Due to the slow evolution of the sensor network topology, OLSR performed satisfactorily for best effort traffic but needed subtle adjustments to balance between latency and bandwidth to meet the requirements of delay-sensitive applications. Lastly, default parameters of OLSR were tweaked and recommendations were made with results that showed promising ways to further improve the performance of OLSR in sensor networks, albeit not as significantly as in the tactical networks for the ship case.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION....................1
A. GENERAL....................1
B. THESIS DESCRIPTION....................2
C. THESIS OUTLINE....................3

II. THEORECTICAL BACKGROUND....................5
A. ROUTING....................5
1. Distance Vector....................5
2. Link State Routing....................5
3. Flooding....................6
4. Source Routing....................6
B. CHARACTERISTICS OF ROUTING PROTOCOLS....................7
1. Single Channel versus Multi-channel Protocols....................8
2. Unicast versus Multicast....................8
3. Uniform versus Non-Uniform Protocols....................8
4. Hierarchical Topology/Clustered Routing....................8
5. Position-based Protocols....................9
C. AD-HOC ROUTING PROTOCOLS....................10
1. Destination Sequenced Distance Vector (DSDV)....................11
2. Ad-hoc On Demand Distance Vector (AODV)....................12
3. Zone Routing Protocol (ZRP)....................14
D. MOBILITY MODELS....................15
1. Random Waypoint....................15
2. Reference Point Group Mobility....................16
E. PERFORMANCE EVALUATION METRICS....................17
1. Packet Delivery Ratio (PDR)....................17
2. Average End-to-end Delay....................18
3. Routing Overhead (ROH)....................18
4. Normalized Routing Load (NRL)....................18
F. CONCLUSIONS AND SUMMARY....................19

III. OPTIMIZED LINK STATE ROUTING PROTOCOL....................21
A. GENERAL....................21
B. OLSR OVERVIEW....................21
1. Multipoint Relay Selection [38]....................22
2. Tuning OLSR Parameters....................23
C. DISCUSSION....................24
D. SUMMARY....................24

IV. SIMULATION ENVIRONMENT AND SETUP....................25
A. GENERAL....................25
B. COMPUTER ENVIRONMENT....................25
C. NETWORK SIMULATOR-2 (NS-2)....................26

D. MOBILE NETWORKING MODEL 27
1. Mobile Node Model 27
a. Link Layer *27*
*b. Address Resolution Protocol (*ARP*)* *29*
c. Interface Queue (IFq) *29*
d. Media Access Control (MAC) Layer *29*
e. Physical (PHY) Network Interface Layer *29*
f. Radio Propagation Model *30*
2. Antenna Model 31
3. Routing Agents 31
E. DESCRIPTION OF THE OTCL SCRIPT 31
1. Setting Up and Defining the Constants 31
2. Defining Node Properties 32
3. Creating Node Movements 33
3. Finishing Up and Running the Simulation 34
F. MOBILITY AND TRAFFIC MODELLING 34
1. Simulation Time 34
2. Number of Nodes 35
3. Pause Time 35
4. Groups 35
5. Node Speed 35
6. Data Traffic 36
7. Propagation Model 36
G. GENERATING THE MOBILITY FILE 36
1. BonnMotion 36
2. Creating the Mobility File 37
3. Converting to NS-2 Format 38
H. GENERATING THE TRAFFIC FILE 38
I. OLSR INSTALLATION 39
1. General 39
2. Tweaking the OLSR Default Parameters 40
J. OUTPUT TRACE FILE FORMAT 40
K. PARSING THE TRACE FILE 42
L. SUMMARY 42

V. RESULTS AND DISCUSSIONS 43
A. SHIP PLATFORM NODE SCENARIO 43
1. Influence of Mobility Rate 43
2. Influence of Node Density 44
3. Influence of Network Loading 47
B. SENSOR NODE BASED SCENARIO 48
1. Influence of Mobility Rate 49
2. Influence of Node Density 53
3. Influence of Network Loading 54
C. TUNING OLSR PARAMETERS 55
D. RESULTS OF OLSR TUNING 58

1. Optimizing Ship Network Scenario 59
a. Influence of Mobility Rate 59
b. Influence of Node Density 59
c. Influence of Network Loading 59
2. Optimizing Sensor Network Scenario 60
a. Influence of Mobility Rate 60
b. Influence of Node Density 60
c. Influence of Network Loading 65
E. SUMMARY 65

VI. CONCLUSIONS, RECOMMENDATIONS, AND FUTURE WORK 67
A. CONCLUSIONS 67
B. RECOMMENDATIONS 68
C. FUTURE RESEARCH AREAS 70
1. Cross Layering Issues 70
2. MAC Layer Adaptations 70
3. Security 70

APPENDIX A. SAMPLE TCL SCRIPT FILE 73

APPENDIX B. SAMPLE MOBILITY OUTPUT FILES GENERATED BY BONNMOTION SOFTWARE 77

APPENDIX C. SAMPLE TRAFFIC FILE 79

APPENDIX D. SAMPLE TRACE FILE 81

APPENDIX E. JAVA PARSER CODE 83

LIST OF REFERENCES 85

INITIAL DISTRIBUTION LIST 91

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

Figure 1. Protocol class overview 7
Figure 2. Clustered routing topology with 5-cluster. 9
Figure 3. Overview of ad-hoc routing protocols. 11
Figure 4. Propagation of a RREQ 13
Figure 5. Propagation of a RREP 13
Figure 6. A routing zone with radius of 2 hops. 14
Figure 7. The categories of mobility models used in ad-hoc networks. 15
Figure 8. Example of movements using RPGM 17
Figure 9. A multipoint relay selection by node *m*. 22
Figure 10. Flow of events for a Tcl file run in NS 27
Figure 11. The MobileNode object in NS-2. 28
Figure 12. Two-ray ground propagation model vs. packet delivery ratio 30
Figure 13. Results of varying mobility rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 45
Figure 14. Results of varying node density on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 45
Figure 15. Results of varying connection rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 46
Figure 16. Results of varying number of data connections on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 46
Figure 17. Results of varying mobility rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 50
Figure 18. Results of varying node density on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 50
Figure 19. Results of varying connection rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 51
Figure 20. Results of varying number of data connections on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 51
Figure 21. Performance metrics for ship network scenario by varying Hello intervals from one to 10 seconds with Topology Control intervals set at 2.5, 5, and 10 seconds. 56
Figure 22. Performance metrics for sensor network scenario by varying Hello intervals from one to 10 seconds with Topology Control intervals set at 2.5, 5, and 10 seconds. 57
Figure 23. Results of varying mobility rate (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 61
Figure 24. Results of varying node density (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 61
Figure 25. Results of varying connection rate (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 62
Figure 26. Results of varying number of data connections (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 62

Figure 27. Results of varying mobility rate (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 63
Figure 28. Results of varying node density (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 63
Figure 29. Results of varying connection rate (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL..64
Figure 30. Results of varying number of data connections (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL 64

LIST OF TABLES

Table 1. Wireless event trace file. 41
Table 2. Simulation parameters of ship nodes 43
Table 3. Simulation parameters of sensor nodes....................52
Table 4. Summary of recommended parameter value for use of OLSR in the ship and sensor network scenarios....................58

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

For my advisor, Professor John McEachen, in trusting and providing me with all the space to be creative and finish up the work.

My appreciation to Professor Weilian Su for his helpful input and candid feedback.

For my 3-year-old son *Asher*, who, when I was so absorbed with the simulations and writing, always prompted with a one-liner: “Daddy, stop working and play with me?”

For my wife
Jonna

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Research concerning mobile ad hoc networks (MANET) is currently of great interest in the Internet community. The vision of a MANET is a spontaneous network that can be established even if the local infrastructure has been destroyed. As an example, consider a coastal patrol mission with teams engaging in maritime interdiction operations, in which forward crews and commanders need to communicate with each other to coordinate their work. The communications network between hosts is not necessarily directly linked nor does it have a fixed infrastructure. Ad-hoc networks are often autonomous, self-configuring and adaptive, which make them an excellent candidate for many such unique military applications.

The rapid adoption of wireless networking technology in the commercial sector using IEEE 802.11 based WLAN specifications is well-documented. It provides a compelling platform and it is prudent for the military to leverage and extend the capability to its own unique needs. This results in a need to identify a new sort of routing protocol. Existing commercial routing protocols, such as the Open Shortest Path First (OSPF) standard, were never designed to be used in an environment in which the network topology can change rapidly and in which nodes are connected by low data rate, high bit error links. This reinforces the fact that existing commercial protocols specifically developed for the wired infrastructure must be appropriately modified before they are used in the wireless ad-hoc networking environment.

Various ad-hoc routing protocols have been proposed in literature. The efforts are largely fueled by the formation of the MANET working group within the Internet Engineering Task Force (IETF) in 1999, to develop a routing framework for IP-based protocols in ad-hoc networks. Network routing protocols may generally be classified into three types: on-demand or reactive, proactive, and hybrid.

On-demand or reactive protocols only create routes when the source node requests it. When a node requests a route toward another node a route discovery process is initiated within the network. The route discovery will end once a route is found or once

all possible routes are examined. The discovered route will then be maintained until it is no longer valid or desired. Proactive protocols attempt to maintain routes from each node to all other nodes in the network. Proactive protocols are also called table driven protocols as they need to maintain tables for storing routing information. Whenever a change in network topology occurs these changes are propagated through the network by the means of a broadcast or flooding. These updates are vital in order to maintain a consistent view of the network topology. Hybrid routing protocols combine the advantages of proactive and reactive routing. Since the advantages of either approach depend on the characteristics of the network, such as the degree of mobility and node density, it could be beneficial to combine them. But the hybrid protocol can be viewed more as a "routing framework" rather than an independent protocol, as potentially any proactive protocol can be employed for intra-zone routing and any reactive protocol can be employed for inter-zone routing.

This thesis focuses on a performance evaluation of three key ad-hoc routing protocols, namely, Destination-sequenced Distance Vector (DSDV), Ad-hoc On-demand Distance Vector (AODV), and Optimized Link State Routing (OLSR). OLSR is an improvement over the older proactive DSDV protocol and AODV is the classic reactive protocol that is cited in many research papers and earmarked to become a standard. Using computer simulations, the protocols are tested based on two scenarios, namely, tactical networks for ships and sensor-based network nodes. The goal is to evaluate scalability and effectiveness under these scenarios. Four performance metrics are measured by varying the maximum speed of mobile hosts, network size and traffic load, to assess the routing capability and protocol efficiency. In the second part of the simulations, the OLSR protocol is further investigated by tuning its default parameters. The objective is to find out if optimal network performance can be achieved with the appropriate selection of the parameter values, and if it would have different effects on the given scenarios.

The simulator used for the thesis is an open-source network simulator called NS-2. NS-2 is a discrete event simulator targeted for network research and provides support for simulating protocols on conventional networks and wireless networks. As in any network simulation software, it is necessary define the network configuration, mobility

pattern describing node movement, traffic pattern describing data traffic and files describing coordinates for obstacles and pathways to closely resemble the real environment. The simulation output is a data file that needs to be parsed to extract meaningful information for post-simulation analysis concerning the protocol behavior.

The results showed that in an ad-hoc network environment where the mobile nodes are expected to have high group mobility such as the tactical ship networks, AODV is a better protocol to use, as opposed to OLSR and DSDV. Although OLSR demonstrated vast improvement over the older proactive DSDV protocol, it did not perform as well as AODV in an open and dynamic networking environment simulating the littoral theatre. When group mobility is accounted for, in a sparse network where node density is in the order of 20 to 30 ships, AODV is less affected by network fragmentation and is able to sustain a superior packet delivery ratio (PDR) of >90% while maintaining a low route latency of <15ms. This is important for supporting delay-sensitive applications such as digitized voice and video. However, in a situation where the network traffic is heavy, approaching 60% of the overall network throughput, it is necessary to consider putting some quality of service (QoS) protection strategies in a contention based network such as the IEEE 802.11. All the protocols are seen to be affected severely in that situation. The poor performance of OLSR is caused primarily by the relatively high administrative routing overheads, which although not apparent in the observed PDR and average end-to-end delay, could become a challenging issue in actual implementation if not appropriately controlled. This is because if we add the real-world problems associated with propagation channels, any excessive overhead packets would only increase the chances of packet collision and may likely cause more latency issues. While this is unlikely to be a problem for best-effort traffic, it can affect the delay-sensitive applications such as digitized voice and video.

On the other hand, OLSR has withstood the tests and acclaimed accolades to emerge as the best protocol to use in the sensor network based scenario. It has consistently outperformed AODV and most definitely DSDV in all cases. Although the routing overhead issue of OLSR is also evident, in this case, the exception is that even AODV has similar problems due to the high node density and low node mobility network envi-

ronment. AODV failed to perform as well in this scenario especially when the node density is increased. This is due in part to its poor routing strategy in a network that has relatively static topology. It also scaled poorly when the number of nodes in an area is in the region of 100 or more. While the choice of protocol is clear in the sensor network scenario, it necessitates a need to balance between higher bandwidth demands and low latency. Best effort traffic is typically not a problem for all cases considered in this scenario.

The results of the OLSR tuning have demonstrated that it is possible to achieve optimality in routing protocol performance by varying the Hello packet and topology control intervals. This was shown to improve the protocol reactivity to topological changes and link failures, while at the same time not incurring any degradation to the overall bandwidth utilization. The tuning resulted in better optimization of OLSR for the sensor networks than for the ship networks. The improvements are significant and seem to apply to all the tests and performance metrics in the sensor network scenario. Whereas for the ship networks, the marginal improvements made to the PDR and average end-to-end delay are actually offset by the increasing routing overhead in all cases.

In summary, the results provided by the simulations are certainly not an end by themselves, but by constantly improving all possible aspects of the simulation environment, will only provide better confidence in their use, including the transition to experimentation and real implementations.

I. INTRODUCTION

A. GENERAL

An ad-hoc network is often described as a collection of mobile platforms or nodes where each node can move freely and arbitrarily without the benefit of any fixed infrastructure except for the nodes themselves. They are often autonomous, self-configuring, and adaptive, which make them an excellent candidate for many unique military applications. The rapid adoption of wireless networking technology in the commercial sector using IEEE 802.11-based WLAN specifications is evidently seen. It provides a compelling platform where it is only prudent that the military would leverage and extend the capability of such wireless solutions to respond to its unique needs. This also results in a need to identify a new class of routing protocols.

Existing commercial routing protocols, such as the Open Shortest Path First (OSPF) standard [1], which uses periodic hello messages to determine network connectivity, were never designed to be used in an environment in which the network topology can change rapidly and in which nodes are connected by low data rate, high bit error links. It reinforced the fact that existing commercial protocols specifically developed for the wired infrastructure must be appropriately modified before they are used in the wireless ad-hoc networking environment.

Protocol development efforts for ad-hoc networks are effervescent in the wireless networking research arena. These efforts are largely fueled by the formation of the mobile ad-hoc networking (MANET) working group within the Internet Engineering Task Force (IETF) in 1999, to develop a routing framework for IP-based protocols in ad-hoc networks. The latest among such works is the introduction of Internet draft RFC 3626 [2] or Optimized Link State Routing (OLSR) protocol. The OLSR protocol is an improvement over the older and less effective proactive routing protocol, the Destination-Sequenced Distance-Vector (DSDV) protocol. It uses a different routing technique designed to adapt to a network which is dense and where data transmission is assumed to occur frequently between large numbers of nodes.

It is observed that most routing protocol research studies for MANET are generally focused on performance optimization for mobile nodes that have no constraints over the implementation of IEEE 802.11 physical channel. The nodes are always assumed to have the physical means for an elevated antenna which can efficiently transmit and find routes between communicating nodes. However, such an assumption does not necessarily hold true where rapid ad-hoc deployment of a surveillance mission is concerned. An example is the application of wireless sensor networks, where nodes typically operate autonomously and are very low profile. Therefore, it is useful to understand if the relative merits of MANET-based routing protocols would apply consistently well into the environmental characteristics of sensor networks.

B. THESIS DESCRIPTION

The goal is to carry out a systematic performance study of three ad-hoc network routing protocols using an open-source network simulation tool called NS-2 [3]. The three routing protocols that will be investigated are, Ad-hoc On-Demand Distance Vector (AODV) protocol [4], Destination-Sequenced Distance-Vector (DSDV) [5], and Optimized Link State Routing (OLSR). Both DSDV and OLSR are regarded as proactive routing protocols that utilize a table-driven technique by recording all routes they find between all source-destination pairs regardless of the use or need of such route. The OLSR protocol, however, is relatively new and the key concept used is that of multipoint relays (MPRs). The use of MPRs is to minimize routing overhead by reducing duplicated retransmissions of routing information in the same region. It is an optimization over a pure link state protocol and henceforth expected to perform better in large and dense ad-hoc networks [6].

For the experiments, the latest release of NS-2 (ns-2.29 [7]) is used. NS-2 is a discrete event simulator widely used in the networking research community. In general, the NS-2 installation will include all software extensions – contributed code developed by the Monarch research group in Carnegie Mellon University (CMU) for simulating multihop wireless networks. It contains a detailed model of the physical and link layer behavior of a wireless network based on the 802.11 specifications and allows arbitrary movement of nodes within a network area. The AODV and DSDV protocols are also provided as part of the NS-2 installation. The simulation for OLSR is implemented using 3rd party

software called UM-OLSR, Version 8.8.0 that is developed by the University of Murcia, Spain [8].

The first objective is to benchmark the performance of the routing protocols based on the technical characteristics of conventional platform-based communication nodes such as a naval patrol vessel. In the context of mobility, a group mobility model is used to simulate movements of military tactical networks. The second objective is to evaluate the scalability of these routing protocols and their performance by extending the simulations to model for higher density and very low-profile mobile nodes. Lastly, we attempt to improve the Quality-of-Service (QoS) of the OLSR protocol by tweaking the default parameters of UM-OLSR to ascertain if it would provide better overall performance to our requirements.

Four important performance metrics are evaluated – packet delivery fraction, average end-to-end delay of data packets, routing overhead, and normalized routing load. The first two metrics are the most important for best-effort traffic. The routing overhead and routing load metric evaluate the efficiency of the routing protocol.

C. THESIS OUTLINE

This chapter provides an introduction to the thesis. Chapter II introduces the theory of routing protocols with detailed discussion on the three different routing protocols that are used for this research. Concepts behind the accurate use of mobility and traffic models are explained. The protocol performance measurement metrics are provided. Chapter III is written specifically to review the OLSR routing protocol as an attempt is made to optimize the protocol performance by tuning some of the default parameters defined in RFC 3626. Chapter IV elaborates on the simulation environment and setup. Chapter V discusses the results of the simulation. Chapter VI provides a conclusion to the thesis. Finally, some recommendations are made and future study areas proposed.

THIS PAGE INTENTIONALLY LEFT BLANK

II. THEORECTICAL BACKGROUND

This chapter provides an introduction to the key concepts and theory related to the research project. The notion of routing and its use in an ad-hoc network are explained. These are supplemented with illustrations on the different types of routing protocols that are either used and/or under research in the academic community. A selection of mobility models is discussed. Finally, performance metrics used to evaluate the routing protocols are explained.

A. ROUTING

The design of routing protocols for MANET often has a traditional routing concept as an underlying algorithm. Therefore, to understand the routing principles in MANET, it is necessary to review conventional routing algorithms such as distance vector, link state, flooding and source routing.

1. Distance Vector

The distance vector routing protocol requires that each router simply inform its neighbors of its routing table. For each network path, the receiving routers pick the neighbor advertising the lowest cost, then add this entry into its routing table for re-advertisement. Routing Information Protocol (RIP) is a common distance vector routing protocol. Common enhancements to distance vector algorithms include *split horizon*, *poison reverse*, *triggered updates*, and *holddown* [9]. The RIP specifications in RFC 1058 [10] provide a good discussion of distance vector, or *Bellman-Ford,* algorithms.

2. Link State Routing

Link state routing requires that each router maintains at least a partial mapping of the network topology. When a network link changes state, a notification, called a *link state advertisement* (LSA) is *flooded* through the entire network. All the routers in the network that are notified of the change will re-compute their routes accordingly. The link state technique is generally more reliable, easier to debug, and less bandwidth-intensive than the distance vector. But it is also more complex and more computationally and memory-intensive. OSPF [1] and Intermediate System-to-Intermediate System (IS-IS) are link state routing protocols.

3. Flooding

Flooding is a packet delivery technique to perform broadcast. In simple flooding, any node can initiate broadcasting a packet to all neighbors in a network. Upon reception of the packet, all of the neighbor nodes rebroadcast the packet. Each node re-transmits the packet exactly once. This process continues until all of reachable network nodes receive the packet [11]. Each node will rebroadcast the packet upon its reception. The nodes of the network will attempt to distribute the packet to as many nodes as possible. So, whenever a node receives the first copy of a message, it just re-transmits it, and the process goes on until the message reaches the destination. While the flooding technique consumes high network resources, it often leads to higher packet delivery ratio.

The other variants in broadcast protocol use what is commonly known as scoped flooding [12]. In scoped flooding, packets are re-transmitted by each node less than once. The main objective is to reduce the number of rebroadcasts to avoid problems associated with broadcast storm. Scoped flooding provides better results in topologies where node mobility is low, where the simple flooding would more likely result in unnecessary re-broadcasts. There are various broadcast schemes that can be found in the literature ([13], [14], [15]) which can generally be classified into the following four groups:

- Probability Based Methods
- Area Based Methods
- Neighbor Knowledge Methods
- Cluster Based Method

4. Source Routing

Source routing is a technique whereby the sender of a packet can specify the route that a packet should take through the network. There are two types of source routing, namely, strict and loose. In strict source routing, the sender specifies the exact route the packet must take. This technique is seldom used as it is relatively restrictive in use for networking purposes. The more common approach is loose source routing in which the sender only determines a few hops that a packet must take to reach its destination. Source routing requires that every node knows the whole network topology. This is usually solved by maintaining a routing table of the given network.

B. CHARACTERISTICS OF ROUTING PROTOCOLS

Since research in ad-hoc networking has resulted in a large amount of routing algorithms and protocols, it can be difficult to decide which algorithms are superior to others under certain conditions. For a successful deployment, this is an important problem, since a wrong choice may have a severe impact on the performance. Also providing just any protocol may not be feasible due to the different requirements on hardware and cross-layer interoperability. Furthermore, all devices in an area would need to agree on one method for true ubiquitous access, so scalability becomes a prime concern. In order to help with the decision, this section provides an overview of the advantages, weaknesses, characteristics, and requirements of routing protocols. The different types of routing protocols, which would be used for the research study, will be described according to these characteristics. Figure 1 provides an overview of the protocol classes.

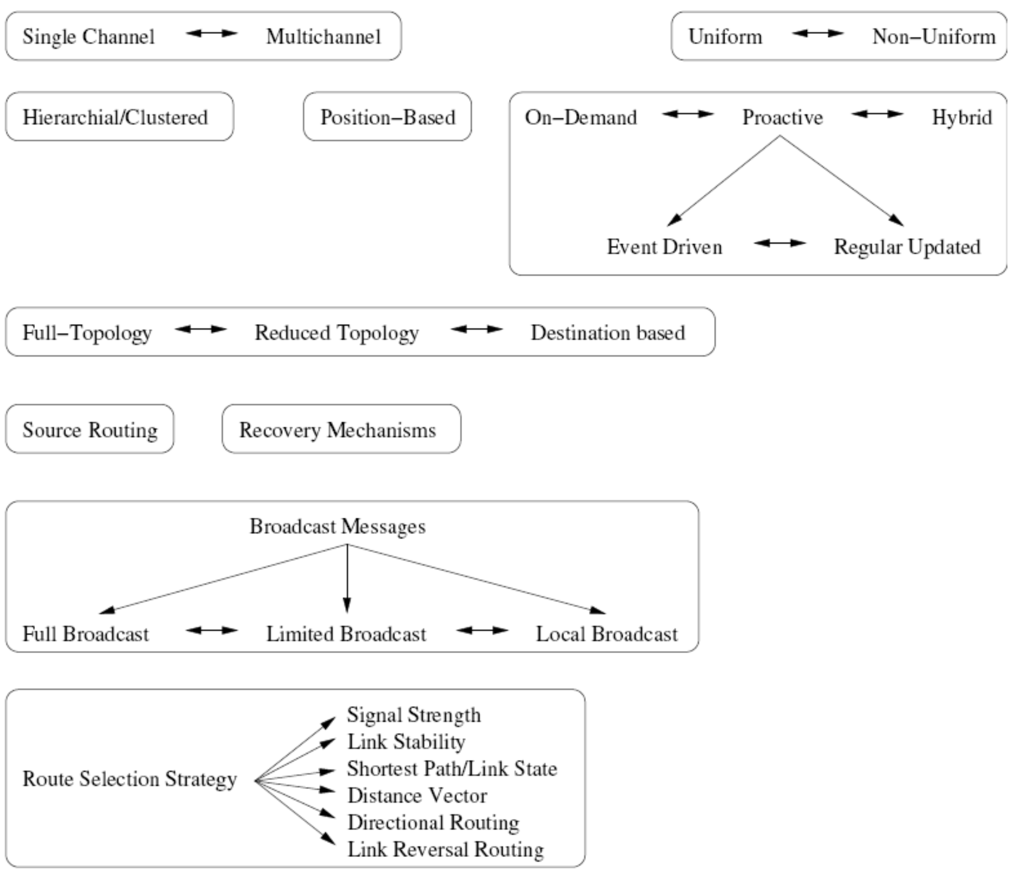


Figure 1. Protocol class overview (Ref. from [16])

1. Single Channel versus Multi-channel Protocols

This property is included for completeness. As in most network simulations, it is often necessary to define a layer 2 interaction and many layer 3 protocols may depend on a certain link layer if they are not designed to operate independently. Single channel protocols use just one shared channel to communicate. An example is the IEEE 802.11 DCF medium access method, which is a widely used example for such a shared channel link layer. Multi-channel protocols typically utilize CDMA, FDMA, TDMA, or its variant form to define some specific channels. Although communication tends to be more efficient using such methods, the problem is always in the implementation. Because multi-channel protocols require a central entity, usually a distinguished controlling station, to assign and manage the channels. Such requirement defeats the purpose of MANET operations but may be viable for sensor networking use. There are some protocols which do not specify the link layer, but their performance may still depend on it.

2. Unicast versus Multicast

With multicast routing, a single packet is sent simultaneously to multiple recipients, as opposed to unicast routing where a single packet is only sent to one recipient on every transmission. The multicast method is very efficient and a useful way to support group communication when bandwidth is limited and energy is constrained. Due to the broadcast characteristics of the multicast protocol it is better suited for MANET than the unicast protocol.

3. Uniform versus Non-Uniform Protocols

A uniform protocol does not assign any special roles to any node. In a non-uniform protocol some nodes may be assigned a special role, which needs to be performed in a distributed fashion. Clustering protocols are typically non-uniform.

4. Hierarchical Topology/Clustered Routing

Clustering is often discussed in the context of ad-hoc networks. Instead of random disposition of networks, clusters are used to introduce some structure into the network. There is usually a dedicated node selected as a group leader, also called a cluster-head. The cluster head forms the cluster and attached nodes use it to describe the cluster they belong to. Clustering in a hierarchical topology has multiple layers of clusters. The cluster-heads are usually responsible for managing communication within a cluster and

are informed about joining and leaving nodes. An example implementation of clustered routing is Cluster-Head Gateway Switch Routing (CGSR) protocol [17]. In CGSR, gateway nodes are also introduced and are used to transmit information from one cluster to another and therefore may be part of more than one cluster. Figure 2 illustrates an ad-hoc network divided into 5 clusters using CGSR.

However, there are known drawbacks to cluster networking, especially with very stable clusters. Since the cluster-heads and also the gateway nodes have to do the routing and management work, they can easily become a bottleneck. Another key issue is that the energy consumption of the cluster-heads or gateway nodes is expected to be much higher than an ordinary node, which can lead to an early outage of these nodes due to limited batteries.

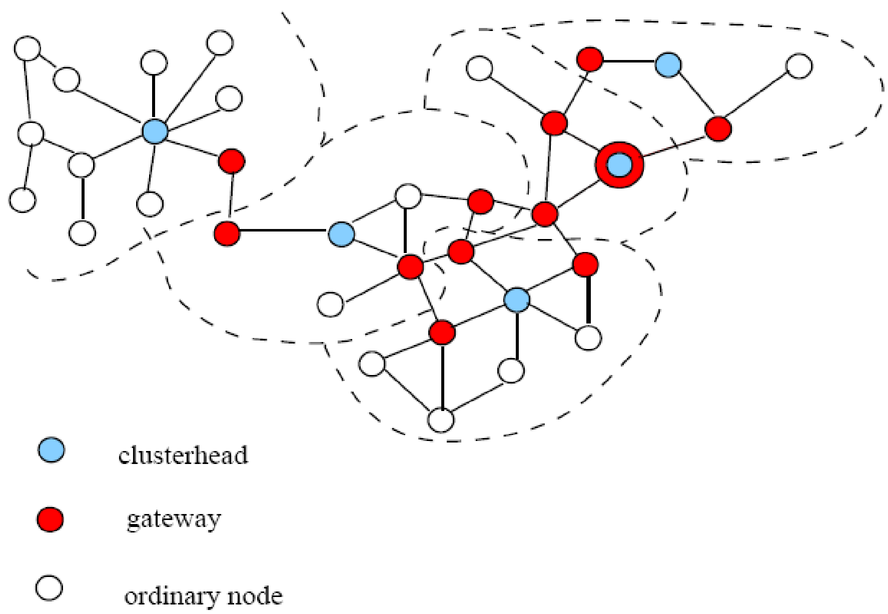


Figure 2. Clustered routing topology with 5-cluster (Ref. from [18]).

5. Position-based Protocols

Position-based routing algorithms require no routing tables and thus no overhead due to route discovery and route maintenance. But a key requirement is the need to have position data of their corresponding destinations, either by an internal discovery process, or by an independent position service. As a result of the extra overhead information that

is required to maintain the position information, and the need for a Global Positioning System (GPS) type system, position-based protocols are not viable for military ad-hoc deployment.

C. AD-HOC ROUTING PROTOCOLS

In order for mobile nodes to communicate, they need to agree on what basis the communication will be performed. This is accomplished by the means of a routing protocol. Ad-hoc routing protocols typically have two routing strategies; reactive approach (source initiated, on-demand driven) and proactive approach (table driven). There also exists hybrid routing protocols that integrate both routing strategies. One typical problem with reactive and proactive routing protocols is they have scalability problems with increasing network sizes in the order of a few hundred nodes. In addition, they may not adapt to network conditions as well as a hybrid.

On-demand or reactive protocols only create routes when the source node requests it [19]. When a node requests a route toward another node, a route discovery process is initiated within the network. The route discovery will end once a route is found or once all possible routes are examined. The discovered route will then be maintained until it is no longer valid or not desired.

Proactive protocols attempt to maintain routes from each node to all other nodes in the network [19]. Proactive protocols are also called table driven protocols because they maintain tables for storing routing information. Whenever a change in network topology occurs these changes are propagated through the network by means of broadcast or flooding. These updates are vital in order to maintain a consistent view of the network topology.

A hybrid routing protocol combines the advantages of proactive and reactive routing. It takes advantage of pro-active discovery within the local neighborhood of a node using a reactive protocol for communication between these neighborhoods. Zone Routing Protocol (ZRP) [20] is one such implementation and some of its key routing strategies are described in a later section.

Figure 3 shows a list of the current routing protocols compiled by Halvardsson and Lindberg in [21]. The protocols are categorized in accordance with the different

characteristics as defined earlier.   Protocols that do not fit into the classification are placed under “other”.   There are security protocols that address various aspects of security threats. Some of these are standalone protocols while others would work together with a routing protocol, all depending on the functionality. Details of the protocol security is given in [21] and will not be further addressed in this thesis.

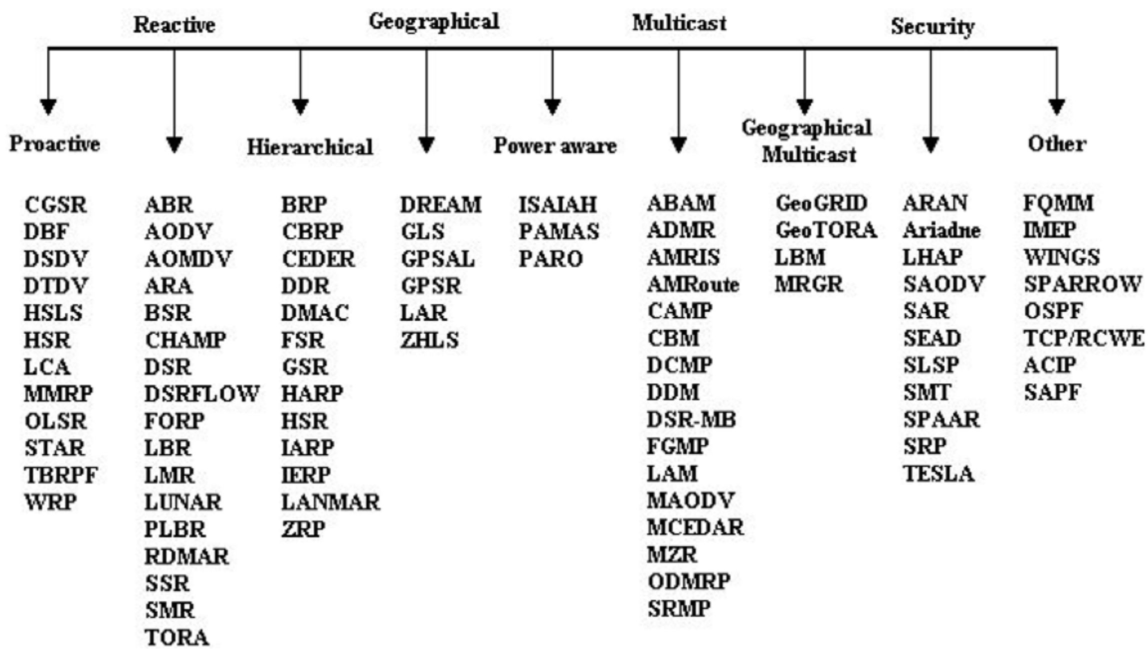


Figure 3.   Overview of ad-hoc routing protocols (Ref. from [21]).

There are many categories and different aspects to take into account when selecting a protocol. In this thesis, one protocol from both the proactive and reactive categories, i.e., DSDV and AODV, are selected for the comparative performance analysis. OLSR is also included given the general acceptance of the protocol being better than DSDV and also better suited for tactical military network operating environments [22].

1. Destination Sequenced Distance Vector (DSDV)

DSDV protocol is the result of adapting an infrastructure network-based distance vector routing algorithm in Bellman-Ford [23], as used in RIP [10], to an ad-hoc networking environment. It improves on the Bellman-Ford algorithm by making sure it is free of loops or routing loops. This is accomplished by assigning each route a unique sequence number. Each routing table lists all destinations with their current hop count and

a sequence number. Routing information is broadcast or multicast. Each node transmits its routing table to its neighbors. Routes with more recent sequence numbers obsolete older routes. This mechanism provides loop freedom and prevents stale routes. The routing information is transmitted every time a change in the topology has been detected (i.e., a change in the set of neighbors of a node). DSDV works only with bidirectional links and would incur excessive routing overheads due to periodic and triggered updates.

DSDV was presented in [5] in 1994. A more detailed description is available in [24]. DSDV was also used for many comparisons like [15], [16] and [25]. The results are mixed, but results from the later papers generally show that DSDV does not perform well compared to the other pro-active protocols. The DSDV simulator is provided as part of NS-2 installation. It is chosen for the simulation primarily for validation as a baseline protocol, as well as for benchmarking with the newer OLSR routing protocol.

2. Ad-hoc On Demand Distance Vector (AODV)

AODV is one of the most discussed and widely used ad-hoc routing protocols. It is an important part of the work of the MANET IETF working group. The draft has reached version 13 and has been submitted to enter the RFC track as an experimental standard. AODV is very well researched and often used as a reference to compare other routing protocols.

The protocol is based on DSDV as described in [5]. AODV limits the number of required broadcasts by only requesting routes when needed, as opposed to DSDV that requires routes to be continuously updated. In AODV, nodes that are not on a selected path do not maintain routing information nor do they participate in any periodic routing table exchanges.

Nodes can be made aware of their neighbors in two ways. When a node receives a broadcast from another node, that is currently not known, it includes this neighbor in its local connectivity information. However if a node has not been active in the ad-hoc network, it can make its neighbors aware of its presence by broadcasting a hello message; these broadcasts are done in a periodic manner. The Hello messages are specified to use a time-to-live (TTL) value of 1, which means that the message will only be broadcasted one hop away.

If a route to a destination is unknown, a route discovery process is initiated. This consists of broadcasting a Route Request (RREQ) packet throughout the network. To limit the impact of a network wide broadcast, these request should be sent with an expanding ring search technique: the TTL of the packets starts with a small value; if no route has been found, the TTL will be increased and the request will be resent. Each node that re-broadcasts this request, adds its address into a list in the packet. Figure 4 shows how a RREQ packet is propagated through the network from the source node toward the destination node. In this figure it is assumed that no intermediate nodes know the route from source to destination. If the destination sees the request, it will reply with a unicast Route Reply (RREP) to the source. Each intermediate node may cache the learned routes. Figure 5 shows the route for the RREP packet.

The routing table entries consist of a destination, the next hop toward this destination, and a sequence number. Routes are only updated if the sequence number of the updating message is larger than the existing one. Thus routing loops and updates with stale information are prevented.

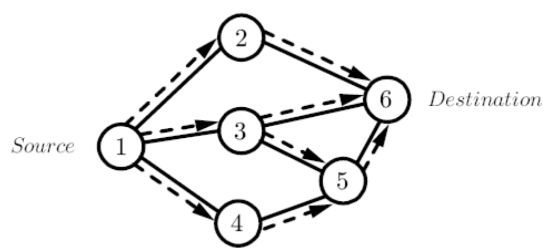


Figure 4. Propagation of a RREQ

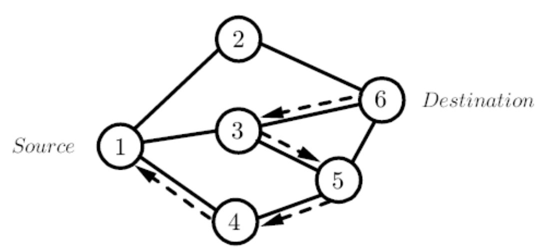


Figure 5. Propagation of a RREP

The authoritative description of AODV is the current IETF draft [26]. There are numerous papers that cite AODV as a reference protocol and it is often demonstrated to provide best results overall, as compared to other on-demand protocols.

3. Zone Routing Protocol (ZRP)

ZRP ([20], [27]) is a zone or cluster-based routing protocol that is a hybrid between proactive and reactive routing. Since the advantages of either approach depend on the characteristics of the network, such as degree of mobility and node density, it could be beneficial to combine them.

ZRP introduces the concept of a routing zone which is a set of nodes within the local neighborhood. In practice, each zone defines a maximum number of hops, and a node within the zone may be distant from the center node of the zone. Each node maintains routing information actively within its zone. The scheme used is called Intra-zone Routing Protocol (IARP), which is essentially an adapted form of a basic link state algorithm. For route discovery outside the local routing zone, a reactive protocol, namely, Inter-zone Routing Protocol (IERP) is used. For this purpose, a bordercast of a request message is used. Bordercast is a technique where the packets are forwarded only to the defined peripheral nodes of the zone. The receiver of bordercast can check if the target is within their own zone, or it may continue to bordercast. The bordercast process is able to identify and reject repeated packets which are already propagated in the same region through recording queries for some pre-defined time at the relaying nodes. ZRP uses a special technique for this, called *Advanced Query Detection* and *Early Termination* [27]. Route caching and local repair are also possible.

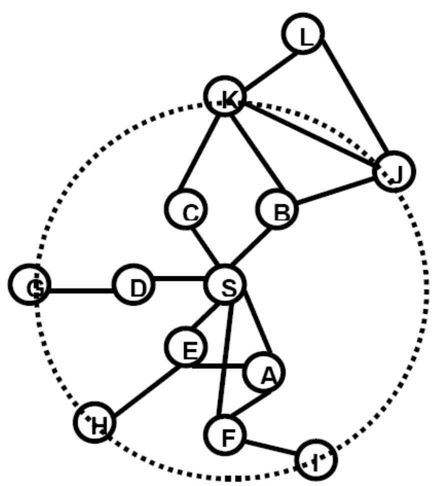


Figure 6. A routing zone with radius of 2 hops (Ref. from [27]).

ZRP is targeted for very large and dense networks, typically in the order of a few hundred nodes, where it is logical to divide the network into zones or clusters of nodes [28]. Such cluster-based routing is not new. A number of cluster-based approaches have appeared in the past [29] [30].

D. MOBILITY MODELS

This section provides an overview of two mobility models that are used in the simulations of ad-hoc networks in this thesis. The mobility model plays an important role in simulations as it dictates the movement pattern a mobile node will follow. Since routing protocols will be deployed in the real world, it is important to simulate them in an environment that resembles their intended scenario. Figure 7 shows the different categories of mobility models that are considered for ad-hoc network simulations in the research community. The details can be found in [31].

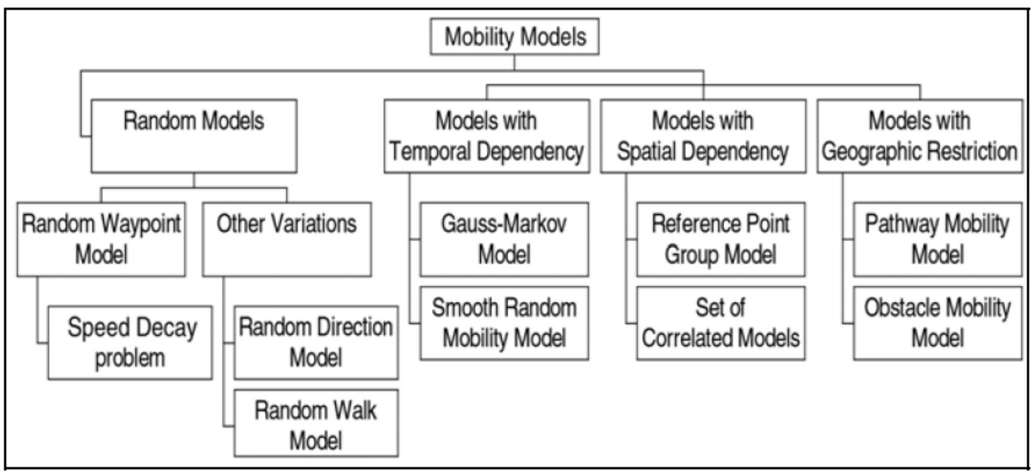


Figure 7. The categories of mobility models used in ad-hoc networks (Ref. from [31]).

1. Random Waypoint

In the random waypoint mobility model, a node would choose at random a destination point and move at a designated maximum speed chosen from [*Vmin; Vmax*], where *Vmin* and *Vmax* are the minimum and maximum speed, respectively. At the destination, the node will pause for a period of time depending on some mobility definition. Upon expiration of the pause time, it would move to a new randomly selected destination within the simulation area.

Random waypoint is basically the same mobility model as random walk if the pause time is set to zero. The random walk model uses principles of *Brownian Motion*. The movement pattern generated by the random waypoint is also, as in the random walk, considered unrealistic. As discussed in [32], a problem that arises in the random way-point model is that the mobile node has a high probability of traveling through the middle of the simulation area to reach their destination. The presence of such phenomena tends to affect the accuracy of examining protocol performance. Random waypoint also suffers from average nodal speed decay. This problem arises because the mobility model fails to reach a steady state with respect to average nodal speed. For example, if one uses [0*;* *Vmax*] as distribution for travel speed one would expect the average speed to be *Vmax/*2. However, the result has shown to be less. The solution to this problem is to set *Vmin* non-zero [32].

The random waypoint model is used only as a benchmark in the initial simulations for this thesis because of its simplicity; it is widely used in many papers when performing comparative analysis of routing performance between protocols, and the model is included in the installation of NS-2.

2. Reference Point Group Mobility

In line with the observation that the mobile nodes in MANET tend to coordinate their movement, the Reference Point Group Mobility (RPGM) model is proposed in [33]. The model is well suited for the tactical movements of naval ships which may form different task groups but require working cooperatively, as well as group mobility of sensor fields for deployment of autonomous nodes for surveillance in open waters.

In the RPGM model, each group has a center, which is either a logical center or a group leader node. The movement of the group leader determines the mobility behavior of the entire group. The respective functions of group leaders and group members are described as follows.

The logical centre is called a reference point (RP). At every time tick the mobile nodes choose a new destination within a circular radius of their reference point's destination. In Figure 8 the white circles correspond to the RP and the black circles correspond to mobile nodes. The movement of the reference point from position RP(t) to RP(t + 1)

is calculated by a group movement vector $\overrightarrow{GM}$. The new position for the mobile nodes is calculated by adding a random motion $\overrightarrow{RM}$ vector to RP(t + 1). The vector $\overrightarrow{RM}$ has its direction from the distribution [0, 2π] and its unit length from the distribution [0, r] where r is the maximum radius centered at the reference point [34].

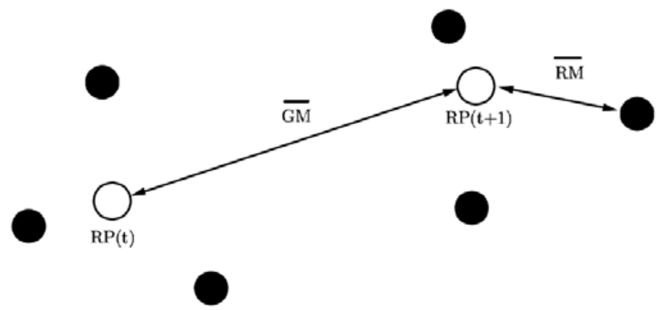


Figure 8. Example of movements using RPGM (Ref. from [34]).

E. PERFORMANCE EVALUATION METRICS

Quality of Service (QoS) is a measurement of how good the routes in the network are. The routes should guarantee a set of pre-specified service attributes such as delivery, bandwidth, and delay variance (jitter). For a protocol to provide good QoS it must determine new routes rapidly and with minimal bandwidth consumption. There are several metrics that directly affect the QoS of every protocol. They include packet delivery ratio, control packet overhead, average hop count, end-to-end latency, and power consumption. Using a protocol that provides good QoS will affect the MANET's performance. The four performance metrics as described below are used in the evaluation.

1. Packet Delivery Ratio (PDR)

Packet delivery ratio is the quotient resulting from the number of unique data packets arrived at the destination divided by the unique data packets sent from a source. Packet delivery ratio measures the protocol performance in the network and this performance may depend on factors such as packet size, network load, as well as the effects of frequent topological changes.

$$\text{Packet Delivery Ratio}=\frac{\sum \text{total packets received by all sinks}}{\sum \text{total packets sent by all sources}}$$

2. Average End-to-end Delay

End-to-end delay is the time it takes for a packet to travel through the network from source to destination. The average end-to-end delay is the summation of all end-to-end delays divided by total data packets arrived at destination.

End-to-end delay is an important routing performance metric since voice and video applications are especially dependent on low latency to perform well. End-to-end delay is to some extent dependent on PDR [28]. That is because if fewer packets are delivered then the average is calculated from fewer samples. As path length increases between source and destination, the probability that a packet will drop also increases. In networks with a low PDR, samples with short paths are favored and thus the delay will be low.

Calculation of the average end-to-end delay includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue, retransmission delays at the MAC, and propagation and transfer times.

3. Routing Overhead (ROH)

The routing overhead is the total amount of control data packets sent by the routing protocol throughout the duration of the simulation. Each time a packet is forwarded over multiple hops, routing overhead is incremented by as many packets as there are hops.

4. Normalized Routing Load (NRL)

Normalized routing load is the number of routing packets transmitted per data packet delivered to the destination. Each hop-wise transmission of a routing packet is counted as one transmission. This metric is also highly correlated with the number of route changes which occurred in the simulation.

The PDR and average end-to-end delay metrics are the most important for best-effort traffic. The routing overhead and the normalized routing load metrics evaluate the efficiency of the routing protocol. Highly efficient routing protocols have lower routing overhead so as to maintain faster route convergence, and thereby, lower overall delay. Protocols with low routing overhead will enable the protocol to scale better and consume less energy. If more control packets are sent by routing agents, the chance of collision

within the transmission channel increases, and thus causes the delay of the application to increase indirectly.

F. CONCLUSIONS AND SUMMARY

The operational movements of naval ships typify a small sized network with moving nodes at high speed and high demands on data delivery. A reactive protocol with multicast capabilities and a mesh-based structure is likely the best choice based on previous literature. The protocol should be able to send information through multiple paths to ensure the throughput of the network is optimized. In situations that demand a high QoS, a protocol that ensures the quality of the network is of great importance. In that case, AODV is a choice protocol and is selected for analysis. It is used to compare against a reactive protocol such as DSDV, as well as the newer OLSR routing protocols. The protocols will be further tested to evaluate their scalability for deployment in a sensor networking type environment, where node density is expected to be higher but with nodes that are very much smaller in height profile.

ZRP is not considered in the suite of protocols to be used for the evaluation due to a few reasons. ZRP can be viewed more as a "routing framework" rather than an independent protocol, as potentially any proactive protocol can be employed for intra-zone routing and any reactive protocol can be employed for inter-zone routing [28]. Also, at the time of this writing, the only ZRP implementation in NS-2 is presented in [35] and is known to work with only the simulator version NS-2.1b6 or earlier. Moreover, the simulation time using the installation of [35] to run ZRP on a dense network is expected to be very long.

This chapter has presented the fundamentals of routing theory and the key characteristics of DSDV and AODV routing protocols. The importance of using the RPGM mobility model for simulations is discussed. The performance metrics are also explained. In the next chapter, an enhanced proactive routing protocol called Optimized Link State Routing (OLSR) is introduced. The design of OLSR and its default routing parameters are detailed.

THIS PAGE INTENTIONALLY LEFT BLANK

III. OPTIMIZED LINK STATE ROUTING PROTOCOL

A. GENERAL

The Optimized Link State Routing (OLSR) protocol was first introduced in [6]. It was later defined as RFC 3626 following a series of Internet drafts published by the IETF MANET working group. Prior to the draft of OLSR Version 5 [36], no public implementation of OLSR was available. The current OLSR Version 11 is the definitive RFC 3626. A few implementations of OLSR can be found on the Internet; the ones found compatible to NS-2 are, namely, OOLSR from Project Hipercom of INRIA (France), NRL-OLSR from the U.S. Naval Research Laboratory, and UM-OLSR [37] from the University of Murcia (Spain). The software extensions of UM-OLSR are chosen for the simulation because the installation was relatively simple and it is compatible to NS-2.29. The software code for UM-OLSR is released under the terms of the GNU General Public License (GPL). This section provides a brief description of the OLSR protocol. The details which are not necessary for the purpose of the thesis are omitted when appropriate.

B. OLSR OVERVIEW

The OLSR protocol is a proactive routing protocol. It provides optimization of a pure link state algorithm tailored to the requirements of a mobile wireless LAN. The concept used in the protocol is that of multipoint relays (MPRs). MPRs are selected nodes which forward broadcast messages during the flooding process. This technique provides two key optimizations [6]. First, it reduces the size of the control packets, that is, instead of all links, it declares only a subset of neighboring links designated as the MPRs. Secondly, flooding of the control traffic is minimized by using only the selected nodes to propagate its messages in the network. Only the MPRs of a node retransmit its broadcast messages. Such procedures substantially reduce the message overhead as compared to pure flooding mechanisms where every node re-transmits each message when it receives the first copy of the packet.

Other than normal periodic control messages, the protocol does not generate extra control traffic in response to link failures and additions. All the routes to destinations are maintained by the protocol; hence it is particularly suitable for networks where a large number of nodes are communicating with each other and where the connections change

with time. The protocol is expected to work well in large and dense networks as the technique of MPRs works well in this context.

The protocol is designed to work in a completely distributed manner and does not depend upon any central entity. It does not require a reliable transmission for its control messages. Each node sends its control message periodically, so it is able to sustain some periodic packet losses, which is typical of radio networks due to collisions, propagation effects, and other transmission problems. Stale route problems are eradicated with sequence numbering, so only the latest information is accepted.

The protocol performs hop-by-hop routing, i.e., each node uses its most recent information to route a packet. Therefore, when a node is moving, its packets can be successfully delivered to it given that it moves together in certain group mobility.

1. Multipoint Relay Selection [38]

Each node *m* of the network independently selects its own set of multipoint relays (MPR) among its one-hop neighbors. The multipoint relay set of *m*, denoted MPR*(m)* is computed in the following manner: it is a minimum subset of one-hop neighbors with a symmetric link, such that all two-hop neighbors of *m* have symmetric links with MPR*(m)*. This means that the MPRs cover (in terms of radio range) all the two-hop neighbors. Figure 9 shows the MPR selection by node *m*. The MPR set is computed whenever a change in the one-hop neighborhood or two-hop neighborhood is detected, i.e., whenever a new one-hop or two-hop neighbor with symmetric link is detected, or a symmetric link with a one-hop or two-hop neighbor has failed.

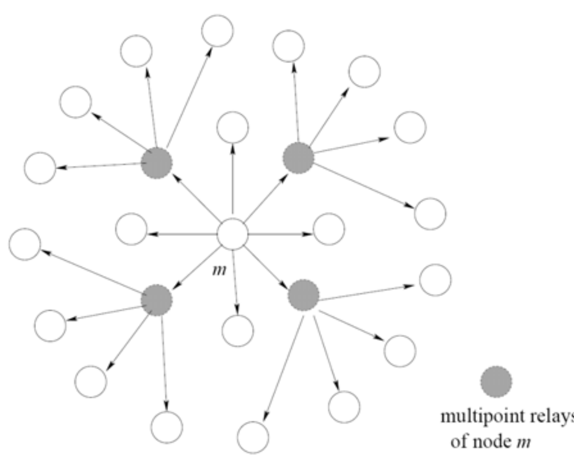


Figure 9. A multipoint relay selection by node *m* (Ref. from [38]).

2. Tuning OLSR Parameters

RFC 3626 defines a set of constants which regulate the usage of OLSR mechanisms. The specification states that parameter configuration may be performed on a per node basis and proposes a set of default values. The following is a summary of the relevant parameters which may affect the network performance and their values are as proposed by default in the OLSR specification.

- HELLO_INTERVAL. This determines the time between sent Hello messages, which is equal to two seconds by default. It is included in Hello messages, so that a node knows the expected time until the next Hello from the same node is received.
- REFRESH_INTERVAL. The list of neighbor interfaces in a Hello message can be partial (e.g., due to message size limitations, imposed by the network), the rule being the following: for each interface a neighbor interface is heard on, its address is cited with corresponding *interface identification* at least once within a predetermined refreshing period, i.e., *Refresh_Interval*. To keep track of fast connectivity changes a Hello message must be sent at least every *Hello_Interval* period, smaller than or equal to *Refresh_Interval*. The *Refresh_Interval*, which is equal to two seconds by default.
- TC_INTERVAL. Topology control (TC) messages can also be partially transmitted like Hello messages. The *TC_Interval* is the period in which the set of advertised links must be complete. It is set to five seconds by default.
- NEIGHB_HOLD_TIME. This indicates for how long the information provided in a Hello message should be considered to be valid. It is equal to 3**Hello_Interval* by default.
- TOP_HOLD_TIME. This parameter is analogous to the previous one, defining the validity period for TC message information. It is equal to 3**TC_Interval* by default.
- WILLINGNESS. This defines how willing a node is to forward traffic. It is specified among a set of eight levels. Its default value is three.

Other parameters that may influence OLSR performance are MPR_COVERAGE and TC_REDUNDANCY. The MPR_COVERAGE allows for definition of the amount of MPRs that should cover a node, whereas the TC_REDUNDANCY tunes the advertised link set in a TC message, and link hysteresis parameters which affect the link establishment and failure procedures.

C. DISCUSSION

As discussed in [39], there is a trade-off between the latency during route change and bandwidth or power consumption, and they are dependent on the OLSR messages sending rate. It was found that reducing the HELLO_INTERVAL and TC_INTERVAL parameters can help improve the protocol reactivity to link failures without significantly degrading bandwidth or power resources considering a certain range of OLSR parameter configurations and scenarios. Assuming that the protocol operates under such conditions, the latency during route change becomes an important performance parameter in a non-static ad-hoc network.

In the UM-OLSR implementation, every UM-OLSR packet may piggyback up to four OLSR messages by default, and the length of IPv4 addresses is used, which can be changed but requires a recompilation of the NS-2 simulator. The other default values of the OLSR parameter are as follow:

- *debug_* : false
- *use_mac_* : false
- *willingness_* : WILL_DEFAULT (3)
- *hello_ival_* : 2
- *tc_ival_* : 5
- *mid_ival_* : 5

The objective is to show that optimization in routing protocol performance can be achieved by characterizing OLSR based on the two types of control messages, i.e., Hello messages and Topology Control (TC) messages.

D. SUMMARY

In this chapter, the general concepts of the OLSR routing protocol was discussed. Its design is unique for an ad-hoc network that is large with high node density. This is ideal especially for sensor networking type applications. In the next chapter, the overall simulation setup is explained as attempts are made to optimize OLSR by tweaking the default parameters.

IV. SIMULATION ENVIRONMENT AND SETUP

A. GENERAL

The simulation software used in this thesis is the network simulator-2 (NS-2) version 2.29. It is open source software and freely distributed under the GNU General Public License (GPL). Generally, the simulations executed in NS-2 do not require large amounts of memory and most of the research work was performed using a Pentium IV computer with 1.86 GHz CPU and 1 GB RAM.

Some of the existing ad-hoc routing protocols are pre-built in NS-2, namely, AODV, DSR, DSDV and TORA. The OLSR routing protocol is considered a relatively new implementation, which has to be acquired separately from 3rd parties. The code from UM-OLSR [37] was selected and installed into the NS-2.29 simulator for this thesis.

In the performance evaluation of protocols for an ad-hoc network, the protocols should be tested under realistic conditions including, but not limited to, a sensible transmission range, limited buffer space for the storage of messages, representative data traffic models, and a realistic mobility model. Essentially, two scenarios are used for the simulations, i.e., one for a platform-based ship environment and another for low profile sensor networking conditions.

NS-2 and the details of the simulator are presented in this chapter. Furthermore, the values for specific simulation parameters and the methodology for generating it are explained. Finally, the simulation procedure is described.

B. COMPUTER ENVIRONMENT

NS-2 can be installed either in a Unix (or Linux) or Windows (2000 and XP) environment. For the Windows environment, it is necessary to install a Unix emulator such as Cygwin prior to the installation of the NS-2 software. One disadvantage of performing simulations in the Windows environment is the issue of software stability. Moreover, most 3rd party software extensions which are available as contributed codes to NS-2 are neither available nor well supported for the Windows platform. The simulations conducted for this thesis were done using the Linux Red Hat 9 operating system (OS).

A hard-disk size of at least 30 GB should be allocated for storage of the simulation files. This estimate is based on the fact that a typical simulation using 50 nodes and 200 seconds simulation time using DSDV and AODV can generate up to 50 MB of trace and NAM files separately. For DSDV and AODV, the actual duration of computer run time is generally between 3 to 20 minutes. In the worst case, the UM-OLSR simulation for 100 nodes can take up to eight hours with a trace file size of about 320 MB. Although the actual run time may be reduced substantially by reducing the simulation time, it is not advisable because there is a potential risk that the results may be inaccurate due to insufficient time provision for convergence in network routing.

C. NETWORK SIMULATOR-2 (NS-2)

NS-2 is a discrete event simulator targeted for network research and provides support for simulating protocols on conventional networks and wireless networks. NS-2 is written in C++ and for simulation setup a script language called OTcl (Object Tcl) is used [3].

A network simulation requires that an OTcl script for network configuration, a mobility pattern describing node movement, a traffic pattern describing data traffic, and files describing coordinates for obstacles and pathways. In addition, it is necessary to trace the mobility model used, as well as the type of traffic at which level, i.e., routing, media access control (MAC) or application, into some output files for post-simulation analysis. Depending on the user's purpose for an OTcl simulation script, simulation results are stored as trace files, which can be loaded for analysis by an external application:

- A NAM trace file (file.nam) for use with any NS-2 compliant animator tool.
- A Trace file (file.tr) which has to be parsed to extract useful information.

Trace file parsing can be done using the *awk* command (in UNIX and LINUX, it is necessary to use *gawk* for the windows environment) or *perl* scripts. The results can then be analyzed using spreadsheets for plotting graphs. XGraph [40] or TraceGraph [41] are two possible tools that can be used to automate the data parsing but would require some technical proficiency in order to extract the data meaningfully. The programs also do not work well when the trace file gets too large.

Figure 10 shows the flow of events for an OTcl file run in NS-2. Appendix A contains a sample of the OTcl script that is used for the simulations in this thesis. The constructions of the script are explained in the proceeding sections.

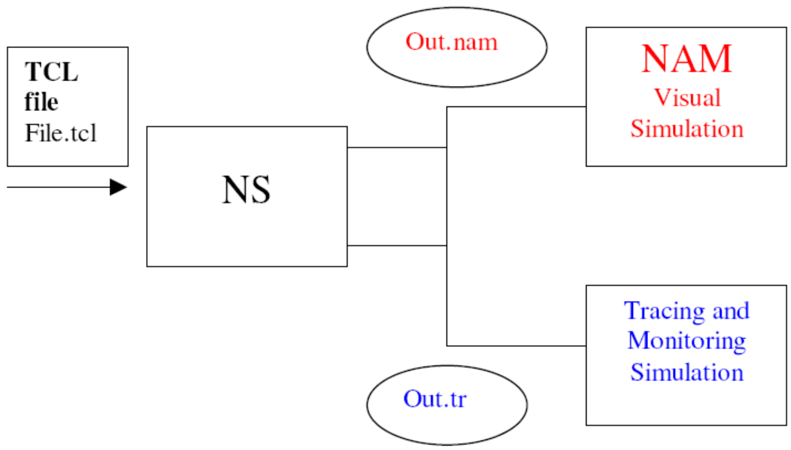


Figure 10. Flow of events for a Tcl file run in NS

D. MOBILE NETWORKING MODEL

The wireless networking model essentially consists of the *MobileNode* class at the core with additional supporting features as shown in figure 11. It is derived from the basic *Node* class. The basic *Node* object provides the added mobility features like node movement, periodic position updates, maintaining topology boundaries, etc.

1. Mobile Node Model

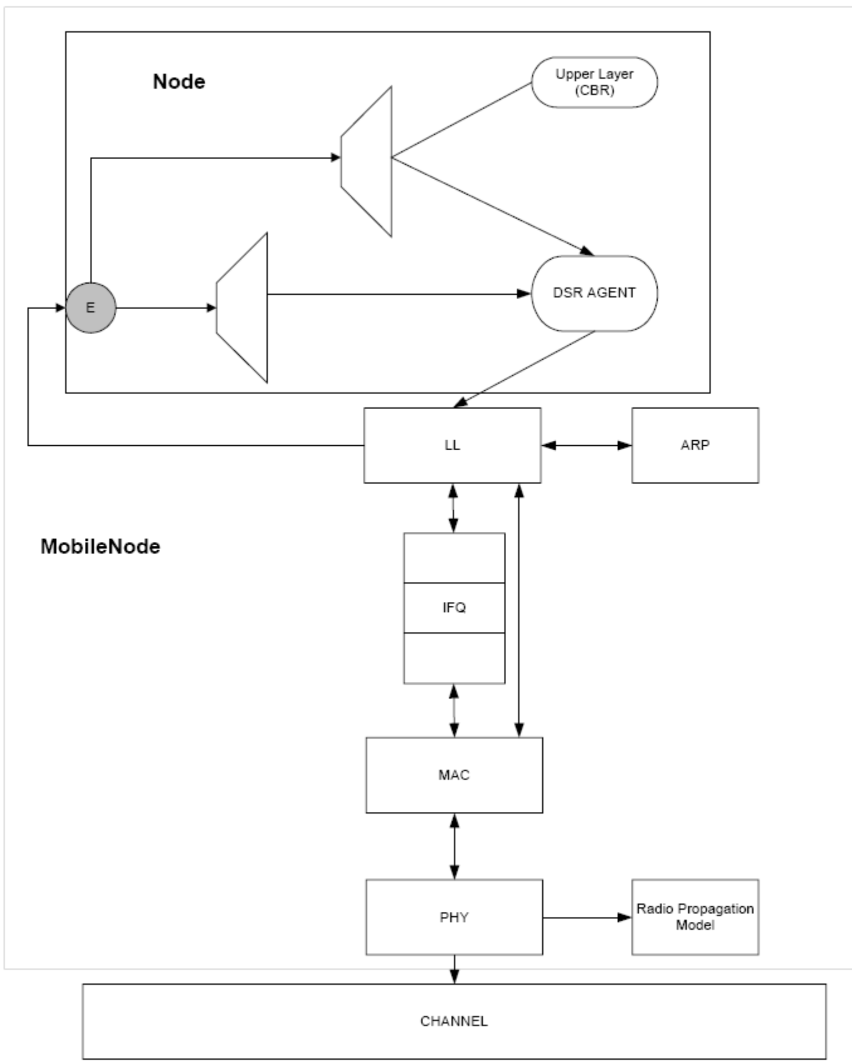
MobileNode is a split object. In addition to the basic *Node* model, it consists of a network stack. The network stack for a mobile node consists of a link layer (LL), an Address Resolution Protocol (ARP) module connected to the LL, an interface priority queue (IFq), a media access control (MAC) layer, a network interface (netIF), all connected to a common wireless channel. These network components are created and plumbed together in OTcl. A packet sent down the stack flows through the LL (and the ARP), the IFq, the MAC layer, and the physical layer. At the receiving node, the packet then makes its way up the stack through the MAC, the LL, etc.

a. Link Layer

The link layer (LL) can potentially have functionalities such as queuing and LL retransmission. The LL object implements a particular data link protocol, such as Automatic Repeat Request (ARQ). By combining both the sending and receiving func-

tionalities into one module, the LL object can also support other mechanisms such as piggybacking.

The LL for mobile node has an ARP module connected to it which resolves all IP to hardware (MAC) address conversions. Normally for all outgoing (into the channel) packets, the packets are handed down to the LL by the Routing Agent. The LL hands down packets to the interface queue. For all incoming packets (out of the channel), the MAC layer hands up packets to the LL which is then handed off at the *node_entry_* point.



C: Classifier LL: Link Layer IFQ: Interface
Queue ARP: Address Resolution MAC: Medium Access Control PHY : Network Interface

Figure 11. The MobileNode object in NS-2.

b. Address Resolution Protocol (ARP)

The ARP (implemented in BSD style) module receives queries from the link layer. If the ARP has the hardware address for destination, it writes it into the MAC header of the packet. Otherwise it broadcasts an ARP query, and caches the packet temporarily. For each unknown destination hardware address, there is a buffer for a single packet. And in case additional packets to the same destination are sent to the ARP, the earlier buffered packet is dropped. Once the hardware address of a packet's next hop is known, the packet is inserted into the interface queue.

c. Interface Queue (IFq)

The Interface queue is implemented as a priority queue, which gives priority to routing protocol packets, inserting them at the head of the queue. It supports running a filter over all packets in the queue and removes those with a specified destination address.

d. Media Access Control (MAC) Layer

Depending on the type of physical layer, the MAC layer must contain a certain set of functionalities such as: carrier sense, collision detection, collision avoidance, etc. Since these functionalities affect both the sending and receiving sides, they are implemented in a single MAC object. For sending, the MAC object must follow a certain medium access protocol before transmitting the packet on the channel. For receiving, the MAC layer is responsible for delivering the packet to the link layer.

The IEEE 802.11 distributed coordination function (DCF) MAC protocol is implemented as part of NS-2. It uses a request-to-send (RTS)/clear-to-send (CTS)/DATA/ACK pattern for all unicast packets and simply sends out data for all broadcast packets. The implementation uses both physical and virtual carrier sense.

e. Physical (PHY) Network Interface Layer

The Network Interface layer serves as a hardware interface which is used by mobile node to access the channel. The interface is subject to collisions and the radio propagation model receives packets transmitted by other node interfaces to the channel. The interface stamps each transmitted packet with the meta-data related to the transmitting interface like the transmission power, wavelength etc. This meta-data in the packet header is used by the propagation model in the receiving network interface to determine

if the packet has minimum power to be received and/or captured and/or detected (carrier sense) by the receiving node. The model approximates the Direct Sequence Spread Spectrum (DSSS) radio interface using Lucent's Proxim's Orinoco (WaveLan) DSSS system.

f. Radio Propagation Model

These models are used to predict the received signal power of each packet. At the physical layer of each wireless node, there is a receiving threshold. When a packet is received and if its signal power is below the receiving threshold, it is marked as an error and dropped by the MAC layer.

NS-2 supports the free space model (at near distances), two-ray ground reflection model (at far distances), and the shadowing model (includes fading). In order to more accurately model the fading environment, an addition to the NS-2 network simulator to handle Ricean and Rayleigh fading is developed by CMU in [42]. The implementation of Ricean Fading is based on [43] and provided for download but the software is only available for patch to NS-2 version 2.1b7 or below, and has not been upgraded since. As such the simulations for this thesis are limited to using the two-ray ground reflection model without fading considerations.

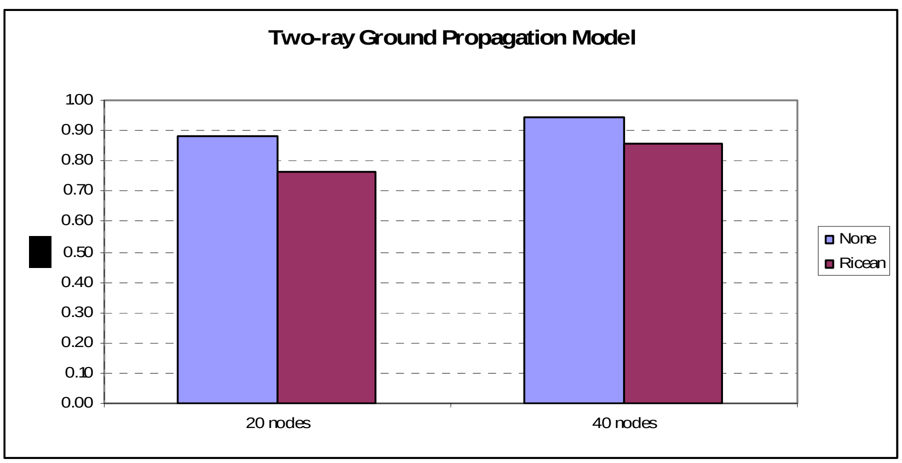


Figure 12. Two-ray ground propagation model vs. packet delivery ratio

A comparative performance between the two-ray model with and without the Ricean fading is provided in Figure 12. The results are obtained by using a separate computer running the NS-2.1b7. The simulations are for 20 and 40 nodes based on a map area of 1000m by 1000m, pause time of two seconds, speed of 10m/s for up to 10 con-

nections. The connection rate is 10 packet per second with packet size of 512 bytes. The performance differentials in packet delivery fractions are shown to be within 10% for up to 40 nodes when the Ricean fading effects (with Ricean K-factor = 6 and maximum node velocity = 2.5m/s) is factored. This may be taken as a guide when looking at the simulation results based on the two-ray model without fading.

2. Antenna Model

The mobile nodes are by default configured with an omni-directional antenna with unity gain.

3. Routing Agents

All packets destined for the mobile node are routed directly by the address de-multiplexer to its port de-multiplexer. The port de-multiplexer hands the packets to the respective destination agents. A port number of “255” is used to attach a routing agent in mobile nodes. The mobile nodes also use a default-target in their classifier (or address de-multiplexer). In the event a target is not found for the destination in the classifier (which happens when the destination of the packet is not the mobile node itself), the packets are handed to the default-target which is the routing agent. The routing agent assigns the next hop for the packet and sends it down to the link layer.

Three different ad-hoc routing protocols will be implemented for the mobile networking. They are DSDV, AODV and OLSR. Both DSDV and AODV routing agents are already integrated into NS-2 as part of the installation. The default parameters for DSDV and AODV are used. For OLSR, the routing agent is implemented by applying the UM-OLSR installation and is described in the following section.

E. DESCRIPTION OF THE OTCL SCRIPT

1. Setting Up and Defining the Constants

The first step in the simulation is to define the wireless physical medium parameters and initialize the simulation. After setting up the initial parameters, the simulator objects are created:

```
#=========================================================
# Set up simulator objects
#=========================================================
# create simulator instance
set ns_ [new Simulator]
```

```
# set topography objects
set wtopo [new Topography]
# create trace object for ns and nam
set tracefd [open $opt(tr) w]
set namtrace [open $opt(nam) w]
# use new trace file format
$ns_ use-newtrace

$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)
# define topology
$wtopo load_flatgrid $opt(x) $opt(y)
# Create God
set god_ [create-god $opt(nn)]
```

The General Operations Director or GOD is a NS-2 simulator object, which is used to store global information about the state of the environment, network, or nodes that an omniscient observer would have, but that should not be made known to any participant in the simulation. The load_flatgrid object is used to specify a 2-D terrain.

The *MobileNode* is designed to move in a three dimensional topology. For simplicity, it is assumed that the *MobileNode* will always move on a flat terrain with Z-coordinate always set to zero. Thus the *MobileNode* has X, Y, Z co-ordinates that are continually adjusted as the node moves.

A flat topology is created by specifying the length and width of the topography, where opt(x) and opt(y) are the boundaries used in simulation.

2. Defining Node Properties

The following API is used to configure the mobile node with all the given values of the ad-hoc routing protocol, network stack, channel, topography, propagation model, with wired routing turned on or off (required for wired-cum-wireless scenarios) and tracing turned on or off at different levels (router, MAC, agent).

```
$ns_ node-config -adhocRouting $opt(adhocRouting) # dsdv/aodv/olsr

        -llType $opt(ll) # specifies link layer object
        -macType $opt(mac) # specifies mac object
        -ifqType $opt(ifq) # specifies ifq object
        -ifqLen $opt(ifqlen) # specifies length of ifq
        -antType $opt(ant) # specifies antenna object
```

```
-propInstance [new $opt(prop)]# propagation object
-phyType $opt(netif) # specifies physical layer object
-channel [new $opt(chan)] # specifies channel object
-topoInstance $topo # specifies topography
-wiredRouting OFF # for wired cum wireless simulations
-agentTrace ON # specifies agent level trace ON/OFF
-routerTrace OFF # specifies router level trace ON/OFF
-macTrace OFF # specifies mac level trace ON/OFF
```

A mobile node is created using the following procedure:

```
for { set j 0 } { $j \< $opt(nn)} {incr j} {
set node_($j) [ $ns_ node ]
$node_($i) random-motion 0              ;#disable random motion
}
```

3. Creating Node Movements

There are two mechanisms to induce movement in mobile nodes. One method is to explicitly define a starting position of the node and its future destinations. These directives are normally included in a separate movement scenario file. Otherwise, the start-position and future destinations for a *mobilenode* may also be set by using the following APIs:

```
$node set X_ \<x1\>
$node set Y_ \<y1\>
$node set Z_ \<z1\>
$ns at $time $node setdest \<x2\> \<y2\> \<speed\>
```

At $time sec, the node would start moving from its initial position of (x1,y1) towards a destination (x2,y2) at the defined speed. With this approach, the node-movement-updates are triggered whenever the position of the node at a given time is required to be known. This may be triggered by a query from a neighboring node seeking to know the distance between them, or the "*setdest*" directive described above that changes the direction and speed of the node.

The second method employs random movement of the node. The following primitive is used to start the mobile node with a random position and have routine up-

dates to change the direction and speed of the node. The destination and speed values are generated in a random fashion.

```
$mobilenode start
```

3. Finishing Up and Running the Simulation

```
# Tell nodes when the simulation ends
#
for {set i 0} {$i < $opt(nn) } {incr i} {
$ns_ at $opt(stop).000000001 "$node_($i) reset";
}

# Tell nam to stop when the simulation ends
$ns_ at $opt(stop) "$ns_ nam-end-wireless $opt(stop)"
$ns_ at $opt(stop).000000001 "puts \"NS EXITING...\" ; $ns_ halt"
puts "Starting Simulation..."
$ns_ run
```

F. MOBILITY AND TRAFFIC MODELLING

Most of the parameters are tweaked within the mobility and traffic files in order to test the behavior of the routing protocols under different performance metrics. Depending on the purpose of the simulations, the parameters have to be set accordingly. It is obvious that some simulation parameters have to be bounded to not result in unduly high computation costs and time.

There are two different scenarios which are being simulated. The first scenario is to set the environment variables based on technical characteristics of conventional platform-based communication nodes such as a naval vessel and the ensuing MANET environment. The second part requires variations to the physical, mobility and traffic profiles to more closely simulate the technical characteristics of higher density and low profile nodes to validate against sensor networking type operations.

1. Simulation Time

Most of the simulation times under the nominal ad-hoc networking tests are set to 200 seconds. In the second part of the simulation with dense networks and low profile

nodes with low mobility, the simulation time is reduced to 100 seconds because it would take too long, sometimes up to 8 hours, to generate one simulation run. In all cases, longer simulation times do not show significant differences in the results. A longer simulation time is also observed to have very little contribution to the results obtained.

2. Number of Nodes

For both scenarios, the node density is varied up to 50 and 100 nodes with simulation areas of 1000m by 1000m and 200m by 200m, respectively.

3. Pause Time

For demonstration of high mobility nodes, the pause time is set to two seconds to stress the network and to prevent the group movement from becoming too predictable. In the second scenario, we assume an inherently low mobility for sensor nodes with a maximum speed of only 0.5m/s. The pause time is set to 100 seconds to simulate nodes that moved out of range from the entire group movement making them unusable nodes.

4. Groups

The number of groups in each scenario is simply the number of nodes divided by the predefined group sizes. In the simulations, group sizes used are either five per cluster or in the context sensor networking, all nodes are configured as one group or cluster. For example, when there are 20 nodes, there will be a maximum of 4 groups with group size of five. Each node in a group is only allowed to move to a random destination but within a maximum group distance of 250m from a reference point's destination under the first scenario. This represents an approximate two-hop distance that is necessary for OLSR to operate efficiently. For the sensor environment, as the effective distance for carrier sense is limited to less than 37 meters for a node profile of 0.1 meter, the maximum group distance is set for 50 meters.

5. Node Speed

There are two different distributions of node speeds in the simulations: 5 to 25m/s and 0.5 to 2.5m/s. This represents a speed of approximately 10 to 40 knots per hour for the naval vessel scenario. The low mobility of 0.5 m/s approximates a flow rate of the Gulf streams.

6. Data Traffic

All data sessions will use constant bit rate (CBR) traffic. Each CBR source transmits data at a rate of 10 packets per second up to a maximum of 50 packets per node. The packet size is 512 bytes which yields a maximum transfer rate of 204 kbps per connection. A maximum load of 10 connections will require a bandwidth of 2 Mbps, which represents a 100% load on the fixed data rate of an IEEE 802.11 wireless LAN (WLAN). A maximum data rate of 2 Mbps is set by default on the 802.11b WLAN interface. Although most 802.11b WLAN cards are faster now and support up to 11 Mbps, it is not necessary to change it for the simulation. By fixing the data rate at 2 Mbps it is good enough for the conduct of protocol performance comparisons.

For the traffic model used in the sensor networking scenario, the generated traffic patterns are modified by hand through changing the node connection sequence in the file. Instead of random connection patterns, two specific nodes are designated as sink nodes and all the data connections are initiated with either of the sink nodes as destination. Two sink nodes are created to simulate node redundancy for operational availability. At least 50% of CBR sources created will be destined to either one of the sink nodes.

7. Propagation Model

The two-ray ground reflection model is implemented as the radio propagation model in all the simulations conducted in this thesis.

G. GENERATING THE MOBILITY FILE

1. BonnMotion

BonnMotion is a Java software application which creates and analyzes mobility scenarios. It is developed by [44], where it serves as a tool for the investigation of mobile ad- hoc network characteristics. The scenario files can be exported for NS-2 and another network simulator, such as GlomoSim/QualNet. The mobility models supported are

- Random Waypoint model,
- Gauss-Markov model
- Manhattan Grid model and
- Reference Point Group Mobility model.

2. Creating the Mobility File

There are two ways to provide scripting input parameters for scenario generation. The first approach is to type the parameters through command line, and the second is to have a file containing the parameters. Both methods may also be combined, but note that the command line parameters will override those given in the input file. The key parameters are:

```
-n        # sets the node number
-d        # sets the scenario duration (in seconds)
-i        # sets the additional seconds to skip before beginning of the scenario
-x, -y    # sets the width and height (in meters) of the simulation area
-R        # set the random seed (manually)
-h        # for RPGM – sets the maximum speed
-c        # for RPGM – sets the probability of group change
-p        # for RPGM – sets the pause time
-a        # for RPGM – sets the number of nodes in each cluster
-d        # for RPGM – sets the maximum distance from group leader
```

For example, the following will generate a Random Waypoint scenario with 100 nodes and duration of 900 seconds. An initial phase of 3600 seconds is cut off:

```
bm -f scenario1 RandomWaypoint -n 100 -d 900 -i 3600
```

The implementation of Reference Point Group Mobility (RPGM) model provides a feature to define "dynamic" groups. When a node comes into the area of another group, it can change to the new group with a probability that can be set with "-c <probability>". The feature can be deactivated with "-c 0". This group change probability is set to 0.01 by default, but is turned off since it can be difficult to determine the overall effects it may have on the simulation results. Other default parameters set by the system include minimum speed (0.5m/s), maximum speed (1.5m/s), maximum pause time (60.0sec) and maximum distance (2.5m). So, it is necessary to vary those settings to suit the requirements of the simulation environment. The following is an example script to generate a RPGM mobility file:

```
bm -f scenario1 RPGM  -d 200 –i 1000 –n 10 –c 0 –m 1 –h 10 –p 2 –x 1000 –y
1000 –a 5
```

The scenario is saved in two files, one with the suffix “.params” which contains the complete set of parameters used for the simulation, and another with suffix “.movements.gz” which contains the zipped file of the movement data.

3. Converting to NS-2 Format

The "NSFile" application is used to generate two files that can be integrated into a Tcl script to start an NS-2 simulation via the "source" command. The following command will generate two files with the suffix “.ns_params" and “.ns_movements”, which will be used in the NS-2 OTcl script.

```
bm NSFile -f scenario1
```

Appendix B shows a sample of the RPGM “.params” and “.ns_movements” output files.

H. GENERATING THE TRAFFIC FILE

Random traffic connections of the transmission control protocol (TCP) and constant bit rate (CBR) can be setup between mobile nodes using a traffic-scenario generator script. The traffic generator script is available under ~ns/indep-utils/cmu-scen-gen and is called cbrgen.tcl. It can be used to create CBR and TCP traffics connections between mobile nodes. An example script can be executed as follows:

```
ns cbrgen.tcl –type cbr –nn 20 –seed 1.0 –mc 10 –rate 10.0 > cbr-20-c10r10
```

The following is a description of the parameters:

- -type either CBR or TCP traffic
- -nn is the number of node(s) to be simulated
- -seed is the seed to the random number generator
- -mc is the maximum number of connections (pair-wise)
- -rate is the rate at which one source generates traffic in packets/second

The traffic file generated is stored under filename “cbr-20-c10r10”. An example of the file content is shown in Appendix C.

I. OLSR INSTALLATION

1. General

OLSR does not come with the standard installation of NS-2. A working version of so-called UM-OLSR Version 8.8.0 is obtained from [37] which is compatible with NS-2.29.

In the installation of UM-OLSR, it is necessary to make changes to the C++ and header files in the ns-2.29 directory. Recompilation of the entire NS-2 must be done whenever changes are made to the C++ or header files. In order to update the *make* and *make.in* file, a *make* command has to be executed at the working directory. A test script can be run to validate that the UM-OLSR code is in working order.

UM-OLSR can be used like any other routing agent in NS-2, the primitive for defining *node* properties is used to attach the routing agent to the mobile nodes which are to be created. The script is as follows:

```
$ns_ node-config -adhocRouting OLSR
```

After creating the mobile nodes, the UM-OLSR routing agent can either be configured individually or all at once. The configuration options of UM-OLSR are:

- debug_ : Print debugging messages on stdout
- use_mac_ : Enable link-layer feedback
- willingness_ : Set the willingness to forward data packets for other nodes
- hello_ival_ : Set the interval of HELLO messages transmission
- tc_ival_ : Set the interval of TC messages transmission
- mid_ival_ : Set the interval of multiple interface message transmission

The following script is used to configure all agents with same default parameters. The lines must be inserted before the commands where the nodes are attached to the channel:

```
# OLSR agent - commented lines are default values and used for tuning
Agent/OLSR set use_mac_    true
#Agent/OLSR set debug_     false
#Agent/OLSR set willingness 3
```

```
#Agent/OLSR set hello_ival_ 2
#Agent/OLSR set tc_ival_    5
```

2. Tweaking the OLSR Default Parameters

The default values of every option available for tuning the OLSR routing protocol in UM-OLSR are as follow:

- *debug_* : false
- *use_mac_* : false
- *willingness_* : WILL_DEFAULT (3)
- *hello_ival_* : 2
- *tc_ival_* : 5
- *mid_ival_* : 5

With all other technical parameters of the simulation scenario remaining unchanged, the periodic intervals between Hello messages (*hello_ival*) and the topology discovery control message (*tc_ival*) are varied between 1 to 10 seconds and 2.5 to 5 seconds, respectively. The results can be used to determine if an optimal performance can be achieved for the OLSR routing protocol.

J. OUTPUT TRACE FILE FORMAT

During a simulation a trace file is generated. It serves as a log file for the events that occurred in the simulation. All trace files are generated with the new wireless trace format [7]. Each field that succeeds the first character is separated by an empty space. Table 1 shows the significance and meaning of each field. Each parameter that follows suit begins with a dash and some letters stating its type.

The first letter of a flag with two letters designates the flag type:

- N: Node Property
- I: IP Level Packet Information
- H: Next Hop Information
- M: MAC Level Packet Information
- P: Packet Specific Information

Depending on the packet type, the trace may log additional information. A more detailed trace file format may be found at [7]. An example of a trace file line is shown below:

```
s -t 4.662822442 -Hs 23 -Hd -2 -Ni 23 -Nx 727.87 -Ny 983.73 -Nz 0.00 -Ne -
1.000000 -Nl AGT -Nw _ -ee _ -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 23.0 -Id 25.0 -It
cbr -Il 512 -If 0 -Ii 141 -Iv 32 -Pn cbr -Pi 2 -Pf 0 -Po 1
```

The trace line, in short, indicates that node 23 is trying to send a CBR packet to node 25 at time approximately 4.66 seconds, from source node (Hs) 23 to destination node (Hd) 2. The source node id (Ni) is 23, its X-coordinate (Nx) is 727.87, y-coordinate (Ny) is 983.73, z-coordinate is (Nz) is 0, energy level is (Ne) 1, the trace level (Nl) is AGT and the node event (Nw) is blank. The MAC level information is given by duration (Ma) 0, destination Ethernet address (Md) 0, the source Ethernet address (Ms) is 0 and Ethernet type (Mt) is 0. The IP packet level information like packet id (Ii) 25.0, source address and port number is given by (Is) 23.0, destination address and port number is (Id) 25.0.

Event	Abbreviation	Flag	Type	Value
Wireless Event	s: Send r: Receive d: Drop f: Forward	-t	double	Time (* For Global Setting)
		-Ni	int	Node ID
		-Nx	double	Node X Coordinate
		-Ny	double	Node Y Coordinate
		-Nz	double	Node Z Coordinate
		-Ne	double	Node Energy Level
		-Nl	string	Network trace Level (AGT, RTR, etc.)
		-Nw	string	Drop Reason
		-Hs	int	Hop source node ID
		-Hd	int	Hop destination Node ID, -1, -2
		-Ma	hexadecimal	Duration
		-Ms	hexadecimal	Source Ethernet Address
		-Md	hexadecimal	Destination Ethernet Address
		-Mt	hexadecimal	Ethernet Type
		-P	string	Packet Type (arp, dsr, imep, tora, etc.)
		-Pn	string	Packet Type (cbr, tcp)

Table 1. Wireless event trace file (Ref. from [7]).

K. PARSING THE TRACE FILE

To extract information from the generated trace file, it is possible to use an *awk* or *perl* script. *Awk* and *perl* functions are already pre-installed in Linux Redhat 9.0. But any programming language can be used to do the parsing. A java program is used to analyze the trace files and is attached on Appendix E. The program is adapted from [45] to include the data extraction for OLSR simulations. This file is used to record the packets and compute the following metrics:

- Number of data packets sent
- Number of data packets received by the destination host
- Total number of routing packets
- Normalized routing load
- Packet delivery ratio
- End to end delay

The program will compute and write the outputs to a CSV file which can be imported into a spread sheet to plot performance graphs. The output trace files may also be visualized in network animator (*nam*).

L. SUMMARY

The simulation environment and setup have been described in this chapter. The mobile node configuration script, as well as settings for generating the traffic and mobility files using the BonnMotion software is detailed. The procedures for tuning the OLSR routing protocol default parameters and methods for analyzing the output trace files are described. In the next chapter, the systematic tests are conducted based on the tactical networks for ships and sensor-based network scenarios. An analysis of the results is provided and discussion made.

V. RESULTS AND DISCUSSIONS

A. SHIP PLATFORM NODE SCENARIO

1. Influence of Mobility Rate

For a better understanding of how the mobility rate affects routing protocol performance, the node speed is increased from 5m/s (about 10knots/hr) to 25m/s (about 50knots/hr) in steps of 5m/s with the number of nodes kept constant, i.e., 20 nodes with a group size of five in a one kilometer squared area. Each node has a pause time of two seconds to simulate a high mobility environment. The traffic type is CBR with a 512-byte data packet. The application agent is sending at a rate of 10 packets per second whenever a connection is made. All peer to peer connections are started at times uniformly distributed between zero and 200 seconds. Each data point presented for this simulation is an average of 3 runs with a different starting seed for the random generator when it loads the mobile nodes into the simulation area, each lasting 200 seconds. The default parameters of the 914MHz Lucent WaveLAN DSSS radio interface model are used. The maximum data rate is set at 2 Mbps, and the IEEE 802.11 distributed coordination function (DCF) is used as the MAC layer protocol. Table 2 summarizes the simulation parameters.

Mobility	
Number of Nodes (N)	20
Map Size	1000m x 1000m
Mobility Model	RPGM
Pause Time	2s
Speed	5-10-15-20-25m/s
Simulation Time	200s
Traffic	
Traffic Type	Constant Bit Rate
Connection Rate	10 pkts/sec
# of connections	10
Packet Size	512 bytes
Routing Protocol	
Protocols	AODV, DSDV, OLSR

Table 2. Simulation parameters of ship nodes

Figure 13 depicts the performance metrics as a function of the mobility rates. Figure 13a shows the packet delivery ratio (PDR) for all protocols, which are all >72%.

It can be seen that AODV and OLSR generally performed better than DSDV. In all cases, the PDR of AODV and OLSR are also higher as the maximum node speed is increased. Intuitively, the PDR is expected to drop at very high levels of mobility as more timeouts are expected to expire before a failure link is declared lost, and in part to the time needed to propagate information on topology change across the network. However, the PDR results showed otherwise and the trace files had to be inspected. This results from using the cluster-based mobility configuration with the RPGM model, as most of the traffic connections are found to be made within the clusters (recall that group size of five is defined) and <20% of the traffic is involved in inter-cluster connections. As the ship nodes have a relatively good communications infrastructure, the effective range is about 550 m for a transmit power of about 24.5 dBm (280 mW) with a carrier sense threshold of -78dBm. This provides an explanation to why the inter-cluster connections had little effect on the PDR performance.

The graphs for the routing overhead packets (ROP) and normalized routing load (NRL) metrics have similar shapes, since in these scenarios the PDR is generally very high. As a result, only the level of both curves changes, i.e., NRL is actually ROP divided by the number of delivered data packets. In Figure 13d, although the NRL of OLSR is about four times higher than AODV, the effects on the average end-to-end delay are not significant (see Figure 13c). For the simulation, the traffic load per connection is only 41 kbps with a full load of approximately 410 kbps when there are ten connections, which represents less than 25% of the overall network throughput of 2 Mbps. Therefore, with the group mobility configuration, the routing protocols will provide better results at higher mobility and the PDF will remain stable at >91% from 15m/s onwards.

2. Influence of Node Density

The density of nodes should have a significant influence on the routing protocols performance. In general, low density may cause the network to be frequently disconnected and high density increases the contention, resulting in a low per-node throughput. In the second set of simulations, the number of nodes per simulation area is increased from 10 to 50 nodes with the rest of the simulation parameters remain unchanged. The goal is not to find the optimal density of nodes, but rather to study how the protocols would scale to different node densities.

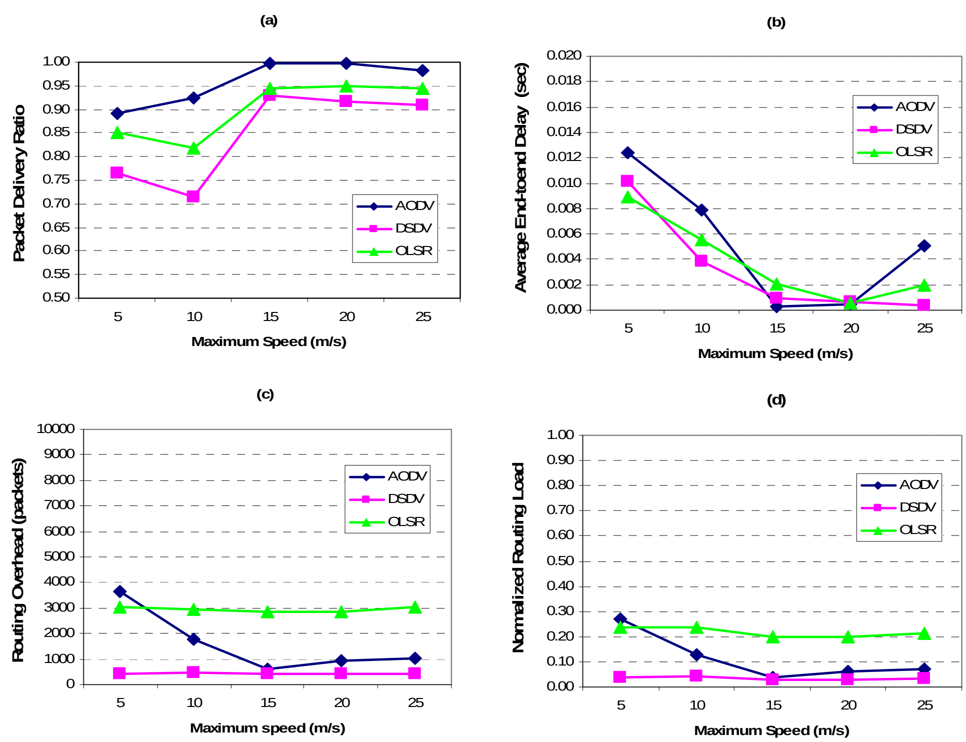


Figure 13. Results of varying mobility rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

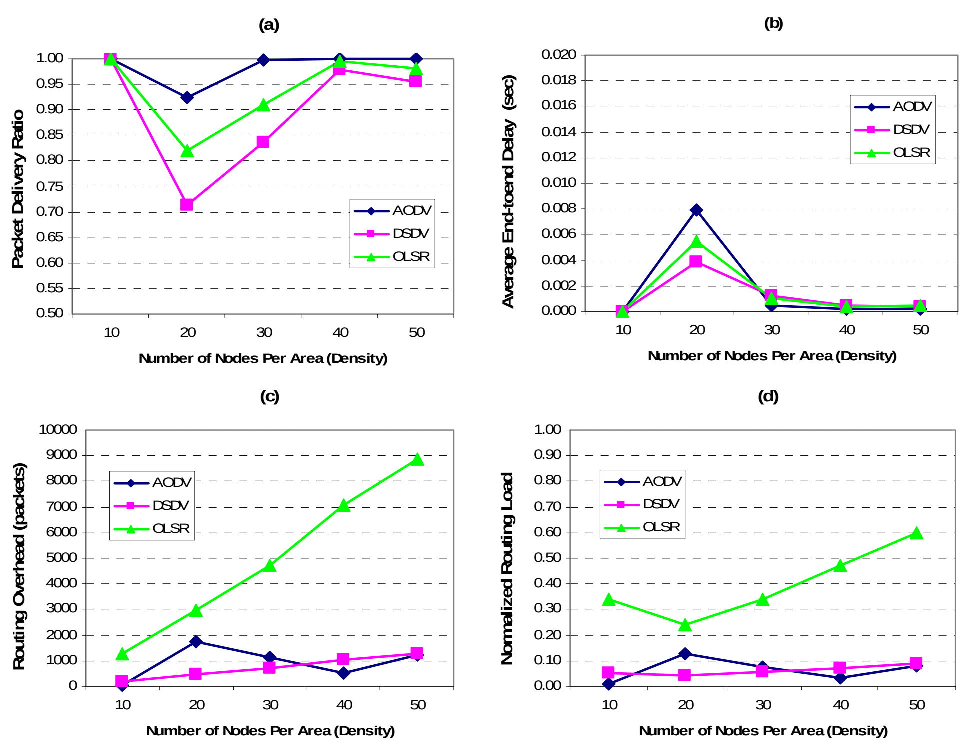


Figure 14. Results of varying node density on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

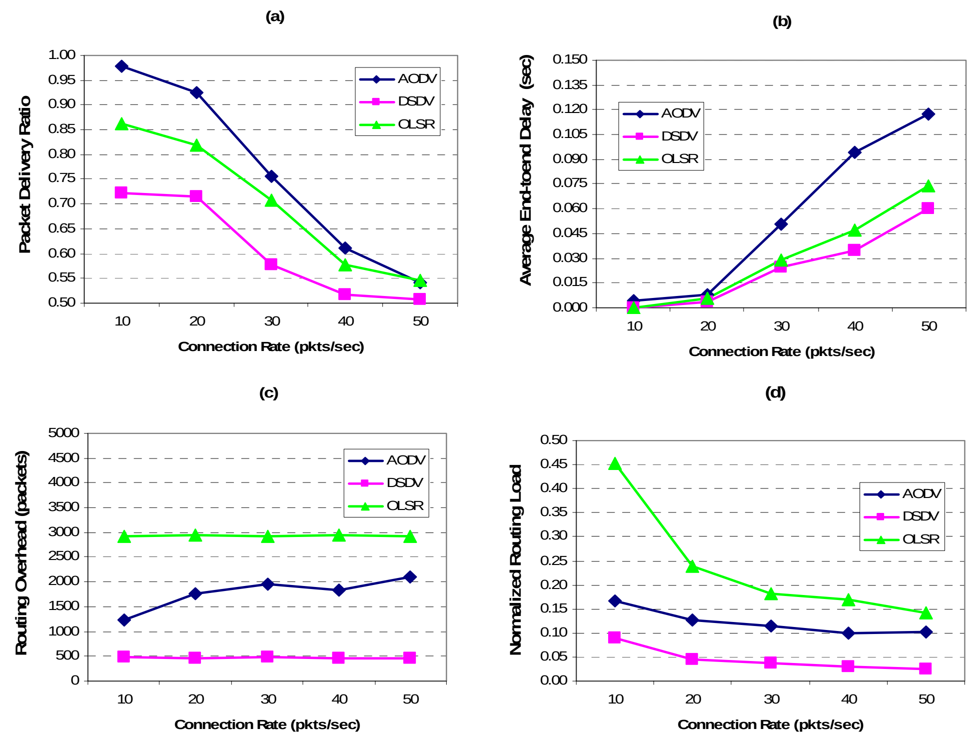


Figure 15. Results of varying connection rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

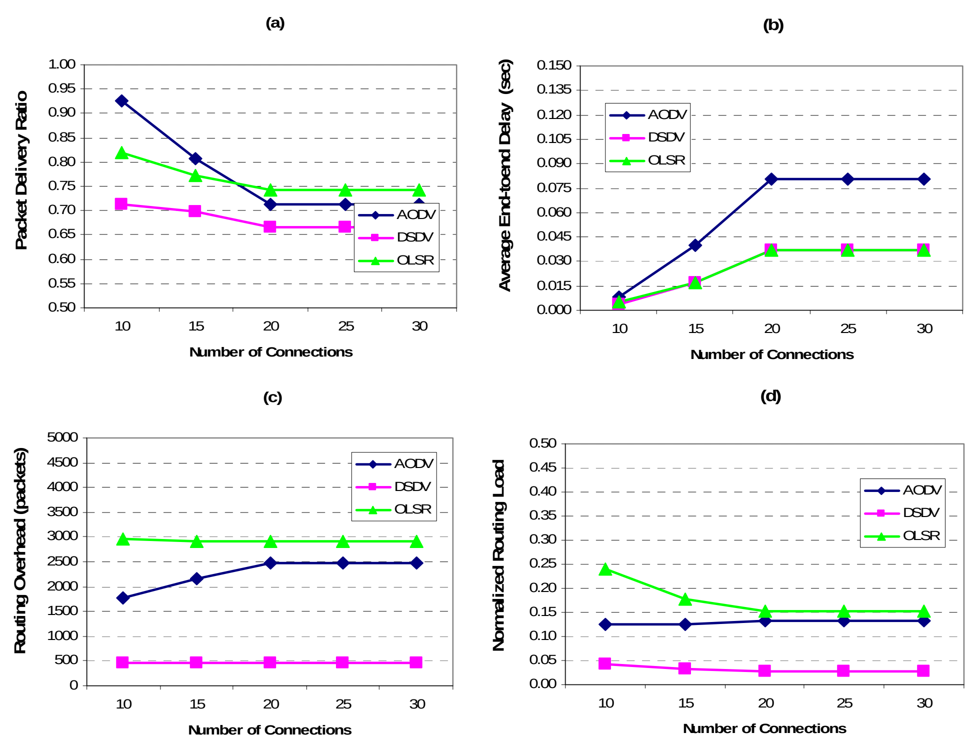


Figure 16. Results of varying number of data connections on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

Figure 14 shows the performance metrics of the protocols as a function of node density. An important observation is that the protocols actually delivered close to 100% of data packets (Figure 14a) despite being in a scenario with a density of only 10 nodes per area, which is not expected in a frequently disconnected network. The reason for this was discussed in the earlier observation of the trace files. Otherwise, the effects of disconnected network due to low node density are evident from the results of 20 and 30 nodes per area, where the PDR improved gradually from around 71% to greater than 91% in all cases as the node density is increased vis-à-vis improvement in network connectivity. However, Figure 14c and 14d revealed a potential problem with OLSR as the routing loads appeared to scale linearly with an increase in node density, although the effect is counteracted by a relatively high PDR and low latency as compared to DSDV and AODV. One of the reasons for the massive amount of routing overheads in OLSR is due to the default setting of the Hello packet and topology control intervals, which are two and five seconds, respectively. For DSDV, the periodic update of routes is set at 15 sec by default. As a result, when the proactive protocols are compared at a node density of 50 nodes, OLSR has effectively six times more routing overhead than DSDV. Secondly, while OLSR utilizes MPRs to reduce routing overhead, the implementation did not perform well as a result of the randomness in node positions as the different clusters or group leaders can move in different bearings. In this case, the MPR flooding method is inefficiently used.

Lastly, as in the case of mobility test, AODV clearly performs better in terms of PDR, while at the same time maintaining a very low routing load and low average end-to-end delay.

3. Influence of Network Loading

Figures 15 and 16 depict the effects of network loading on the performance metrics by increasing either the connection rate or the number of data connections from 10 to 50 and 10 to 30, respectively.

Figures 15a and 16a show that the PDR for all protocols have a declining trend when the connection rate is increased, although the case is less obvious when the increasing number of data connections. As the node density is low it is more likely to result in

network fragmentation as the protocols react to topology changes. Because the connection rate is high, each link breakage resulted in more dropped packets, which explains the low PDR from a connection rate of 20 pkts/sec onward. Both AODV and OLSR reacted consistently to the two network loading conditions. The latency of AODV increased significantly from 5 ms to 117 ms and 3 ms to 37 ms, whereas the latency of OLSR increased from 2 ms to 80 ms and 5 ms to 37 ms when the connection rate and number of connections were increased. All the protocols appeared to follow an increasing trend in network latency when the network load was increased. This is consistent with the declining PDR in Figure 15a, but Figure 16a showed that the protocols are less affected by the increase in number of connections, and OLSR actually performed marginally better than AODV and DSDV.

From Figure 15c and 16c, the results for DSDV always demonstrated a lower ROP than AODV and OLSR. The advantage is a factor of 5-to-6 times. When either the connection rate or number of connections increases, the NRL for OLSR slowly approaches the routing efficiency of AODV. The NRL indicated a factor of 1.4-to-2 times performance differentials between the routing efficiencies of AODV and OLSR. So, AODV is better than OLSR in that metric. In fact, the margin is bigger when the network load is light. In addition, it can be seen that AODV outperformed OLSR by a factor of up to 5 times in NRL, i.e., when there are 10 connections sending at individual data rate of 41 kbps, OLSR with NRL of about 0.45 will need to send out approximately one routing packet for every two data packets.

B. SENSOR NODE BASED SCENARIO

In this scenario, the objective is to analyze how well the routing protocols scale with the size of a sensor network. The context of a sensor network in this thesis is defined as an autonomous, multi-hop, wireless network with nondeterministic routes over a set of physical layers that can serve numerous sensor applications without manual reconfiguration. The simulation focuses mainly on the constraints of the physical environment on overall routing performance and does not address how the routing behavior may potentially affect the dynamics of what the higher layer applications may impose.

The physical profile of a sensor node is assumed to be no smaller than 0.1m and has infinite energy placed in a simulation area of 200m by 200m. The grid size is made

small to avoid incurring long simulation run-time and in part due to low mobility of the nodes. The mobility is to account for the drifting effect of the sea surface which is not necessarily stationary. Also, instead of limiting the broadcast range of the 802.11 radios to simulate communications range of sensor nodes as suggested in [46], a similar effect is achieved by reducing the antenna height from a default value of 1.5m to 0.1m. As such, traffic connections are established via a generic MAC algorithm rather than an imposed hypothetical bound which limits the number of nodes that can receive any broadcast message that is sent. Another key area that is considered for the sensor networking scenario is that the data traffic is not generated in an N-by-N fashion. Instead, there is a designated *sink* node that will either pull or receive data generated by other sensor nodes. This unique traffic pattern is modeled by modifying the generated traffic files by hand, in which two specific nodes are designated as *sink* nodes. The two sink nodes are created to simulate node redundancy for operational availability and at least 50% of CBR sources created will be destined to either one of the sink nodes. The rest of the simulation parameters are varied accordingly to analyze the effects of mobility, node density, and network loading have on the protocol performance.

1. Influence of Mobility Rate

During this experiment, the number of nodes is kept constant, i.e., there is only one cluster of 50 nodes. The node speed is varied from 0.5m/s to 2.5m/s in steps of 0.5m/s. Each node has a pause time of 100s to simulate a low mobility environment. The traffic type is CBR with 512-byte data packets. The application agent is sending at a rate of 10 packets per second whenever a connection is made. All peer to peer connections are started at times uniformly distributed between zero and 100 seconds. The averaging of the data points is not done for this simulation because of very long simulation run-time with OLSR. The same 802.11 radio interface at a 2 Mbps data rate is used. Table 3 summarizes the simulation parameters.

Figure 17 depicts the performance metrics as a function of the mobility rates. Figure 17a shows a declining trend in PDR for all protocols when the maximum node speed is increased. Both AODV and OLSR outperformed DSDV as expected, at a margin of at least 20% higher in PDR. DSDV is unable to effectively deliver data in a dynamic network populated by a large number of nodes. Since sensor nodes are typically

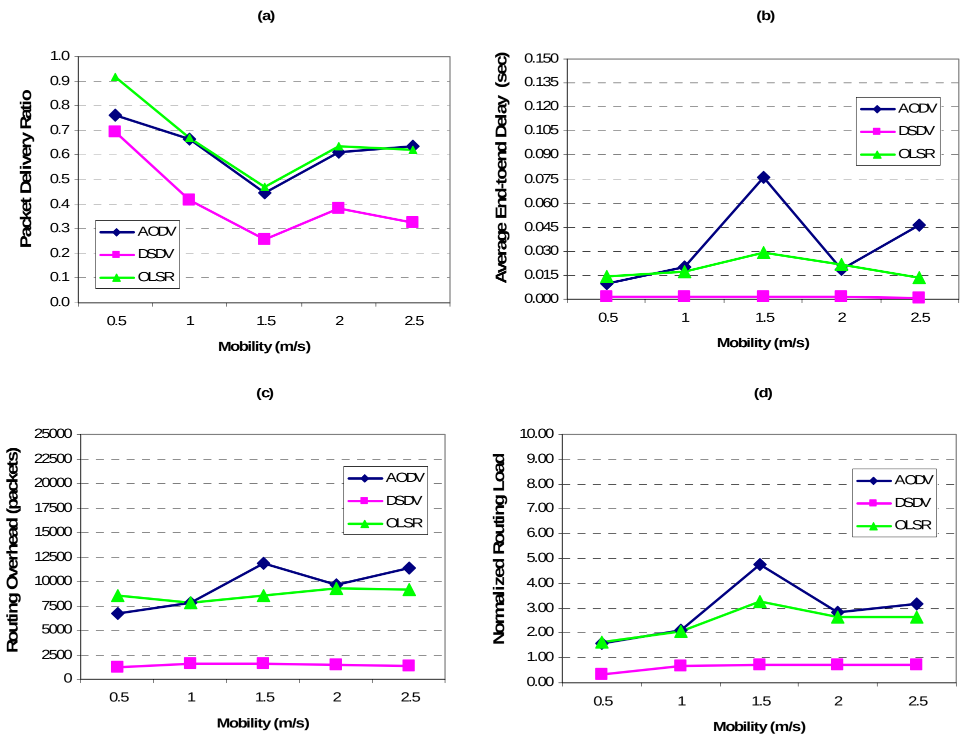


Figure 17. Results of varying mobility rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

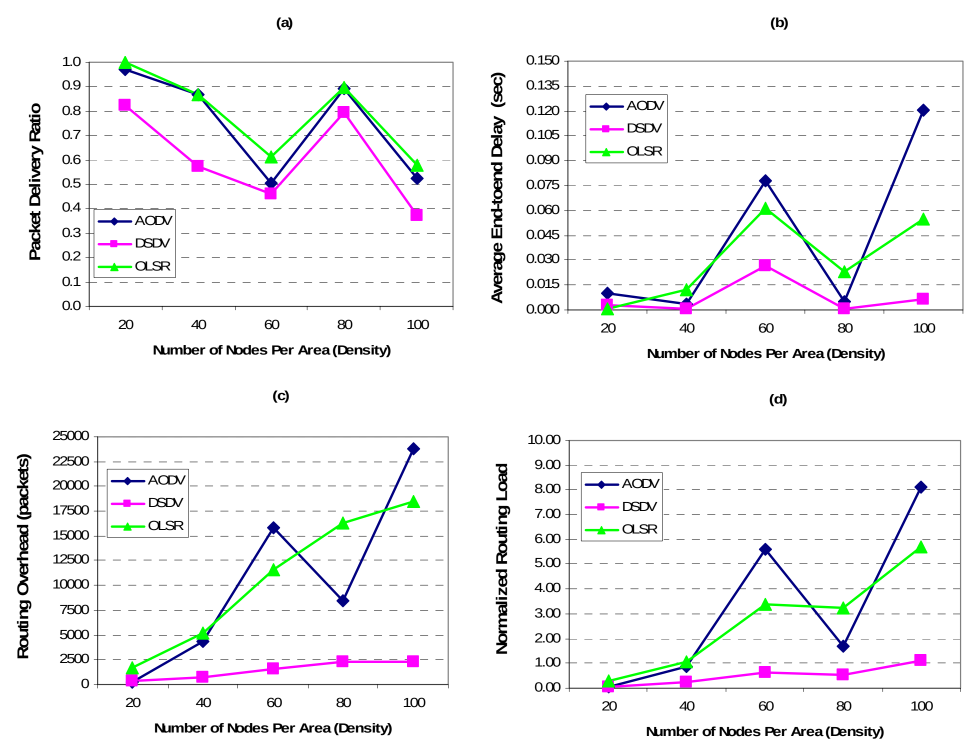


Figure 18. Results of varying node density on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

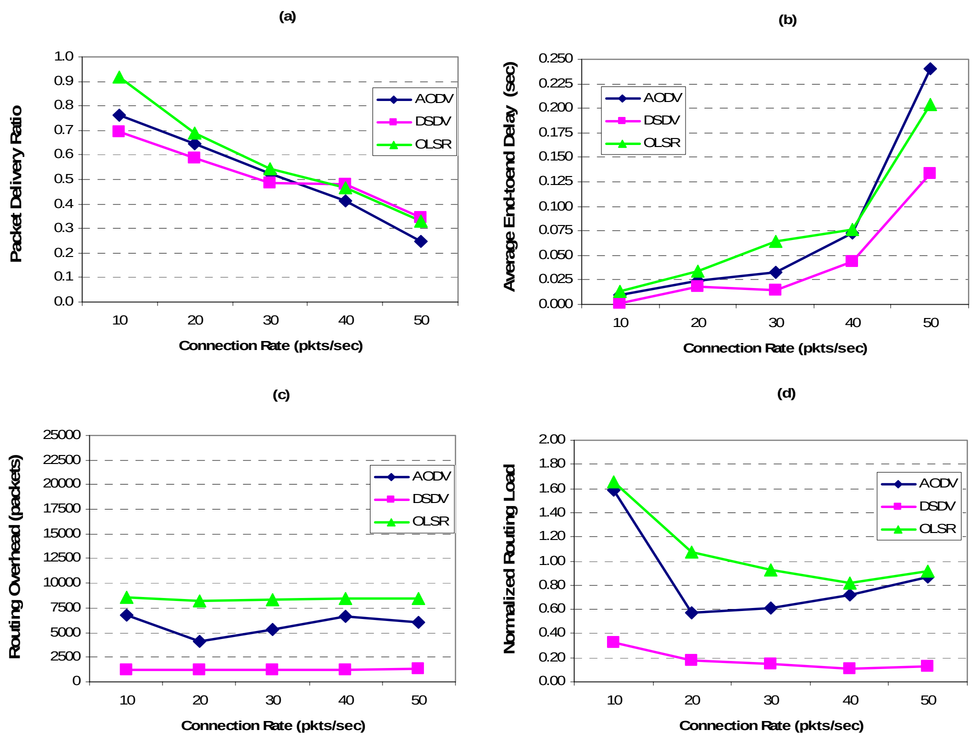


Figure 19. Results of varying connection rate on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

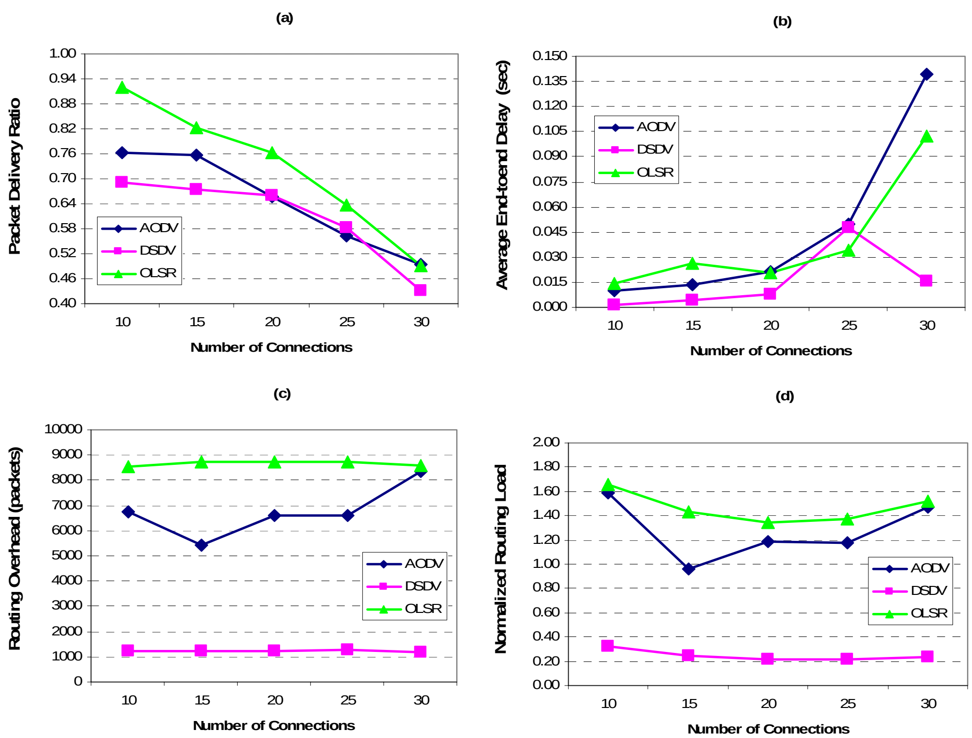


Figure 20. Results of varying number of data connections on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

Mobility	
Number of Nodes (N)	50
Map Size	200m x 200m
Mobility Model	RPGM
Pause Time	100s
Speed	0.5-1-1.5-2-2.5m/s
Simulation Time	100s
Traffic	
Traffic Type	Constant Bit Rate
Connection Rate	10 pkts/sec
# of connections	10
Packet Size	512 bytes
Routing Protocol	
Protocols	AODV, DSDV, OLSR

Table 3. Simulation parameters of sensor nodes

low in mobility, the result is actually more significant at node speed of 0.5m/s or below. At the lowest maximum speed of 0.5m/s, the PDR of OLSR reached the level of 92%, which is significantly better than AODV (76%) and DSDV (69%).

Figure 17b provides a result that is consistent with the theoretical performance capability of proactive protocols in a fixed network. With a network of a large number of nodes with low mobility, the occurrence of link outage is relatively small, so we expect proactive protocols to yield low route latency as compared to reactive protocol. However, in the case DSDV, it is clear that the very low average end-to-end delay is a direct result of the low PDR which indicated a biased preference to short paths with low delays by the protocol. While this may be a good routing strategy for best effort traffic, it will not work well for supporting higher priority traffic such as voice and video, which require a good balance of high PDR and low latency.

The ROH and NRL performance differentials between DSDV and the other two protocols are as explained in Section A of this chapter. The significant point is that OLSR has generally performed better than AODV in all metrics concerning the influence of node mobility on routing protocols.

2. Influence of Node Density

As the scenario is to model a sensor field for this simulation, the number of nodes per simulation area is increased from 20 to 100 nodes with the rest of the simulation parameters remaining unchanged. The effective radio range for 802.11 with 0.1 m node height with two-ray path propagation model is 36.7 meters. So, in the context of sensor networks, the 20 to 100 nodes per area represents a density of μ =5.2 to 10.6.

Figure 18 shows the performance metrics of the protocols as a function of node density. As Figure 18a shows, there is a general down trend of PDR for all protocols when the node density is increased. Although OLSR performs relatively well amongst the three protocols, the result actually contradicts the theory that OLSR will perform better in a large and dense network [6]. It can be seen that at 20 nodes per area, the PDR of OLSR is almost 100% but dropped gradually to 58% at 100 nodes per area. However, a scalability modeling of ad-hoc routing protocols research done in [47] provided a distinctive argument to this. Specifically, in some relatively low density scenarios, the proportion of PDR can increase as more nodes are added because network fragmentation is reduced. But for some high density scenarios, adding nodes can result in good quality N-hop route being replaced by poor quality (N-1)-hop route. This effect is clearly visible in results of this simulation, although it cannot be validated by inspection of the trace files due to complexity and time. For example, the output trace file for OLSR is up to 250Mb for the 100-node simulation, which is too big for manual analysis.

Both AODV and DSDV have similar NRL, which are approximately 0.05 for 20 nodes, whereas the NRL for OLSR is higher at 0.3. AODV and OLSR introduced more ROH as the number of nodes increased, with the load of AODV growing faster than for OLSR (see Figure 18c and 18d). At 100 nodes, the NRLs for AODV and OLSR are 8.1 and 5.7, respectively. That means that for every data packet sent, AODV will require an additional eight routing overhead packets as opposed to about six for OLSR. This is a serious concern because with more packets sent, the chance of collision will be higher in a contention based network. This causes the delay of the application to increase indirectly. Figure 18b provides a proof of this effect. In general, OLSR is more scalable than AODV with respect to the number of nodes per area. It seems that 40 nodes per area is

the turning point. For more than 40 nodes, OLSR performs better than AODV in PDR, NRL and average end-to-end delay.

3. Influence of Network Loading

Figures 19 and 20 depict the effects of network loading on the performance metrics by increasing either the connection rate or the number of data connections from 10 to 50 and 10 to 30 respectively.

Figures 19a and 20a show that the PDR for all protocols have a declining trend when the connection rate is increased. The PDR of OLSR dropped from 92% to 33% when the connection rate is increased from 10 to 50, AODV dropped from 76% to 25%, while that of DSDV dropped from 69% to 34%. The PDR for all protocols dropped more gradually with increasing number of connections. Both AODV and OLSR are able to maintain >50% PDR in the latter case.

As Figure 19b shows, for connection rate of 10 pkts/sec, all the protocols have relatively small latency of <14ms. The latency increases gradually with the connection rate until it reached 40 pkts/sec where the latency of AODV and OLSR increased more rapidly than that of DSDV. The difference in latency between the load at 40 pkts/sec and 50 pkts/sec is about three times, i.e., from 75 ms and 240 ms, which is significant. If the sensor network is to support time-sensitive applications a connection rate of 40 pks/sec, which translates to a data rate of 164 kbps per connection, seems to be the highest throughput that may be used, albeit with penalty of low PDR.

For DSDV, the number of protocol packets appeared to be determined mostly by the mobility and network size. The NRL stayed relatively constant at between 0.12 to 0.32 with increasing connection rates and number of connections (Figure 19d and 20d). The NRL for OLSR is less affected by the network load due to the increase in the number of connections as opposed to increasing connection rates. At either end of the network load OLSR performed better than AODV. Otherwise, the graphs in Figure 19c and 20c indicate that the advantage of a better PDR for OLSR is offset by a poorer average end-to-end delay at medium network loads.

C. TUNING OLSR PARAMETERS

The Optimized Link State Routing (OLSR) protocol is discussed in detail in Chapter III where the methodology of how OLSR finds an alternative route when a link failure takes place is described. Ideally, a good routing protocol should be able to rapidly provide optimal routes. In the case of link failures in an active path, the route to a destination should be re-established with minimum impact on data latency, available bandwidth, and device power consumption for any data traffic pattern. In this section, the OLSR protocol behavior is investigated by tuning some of the default parameters, in particular, the default values for Hello Intervals (Hi) and Topology Control Intervals (TCi). The attempt is to find out if optimal network performance can be achieved with appropriate selection of the parameter values and if it would affect the given scenarios differently.

Based on the default parameters of the ship and sensor network scenarios given in Tables 2 and 3, the simulation is set-up with maximum node mobility of 10m/s and 0.5 m/s respectively. Three graphs are generated in each case by changing the TCi value to 2.5, 5, and 10 seconds while at the same time varying the Hi from one to 10 seconds. The default values of Hi and TCi as proposed in RFC3626 are 2 and 5 seconds respectively.

Figures 21 and 22 show the results of the simulation in terms of the performance metrics for both scenarios. Figure 21a and 22a depict a relatively uniform graph with tight bounds to show that while varying the TCi and Hi would affect the protocol reactivity to link failures, there is generally no impact on the bandwidth performance. The PDRs are seen to drop gradually in the same manner, and decline more rapidly with Hi of >6 seconds. It is obvious that only a Hi range of one to six seconds should be considered in the implementation for both scenarios, for a guaranteed PDR of >82% and >77%, respectively.

Figures 21b and 22b show that the variations in latency is generally not significant (<15ms) in the ship network case whereas the sensor network is very sensitive to changes in both parameters. This is a result of the mobility in a dense network. This is due to the fact that the sensor node has a small communications range and often requires short Hi to maintain its link topology. As a result, it did not react well to long Hi interval

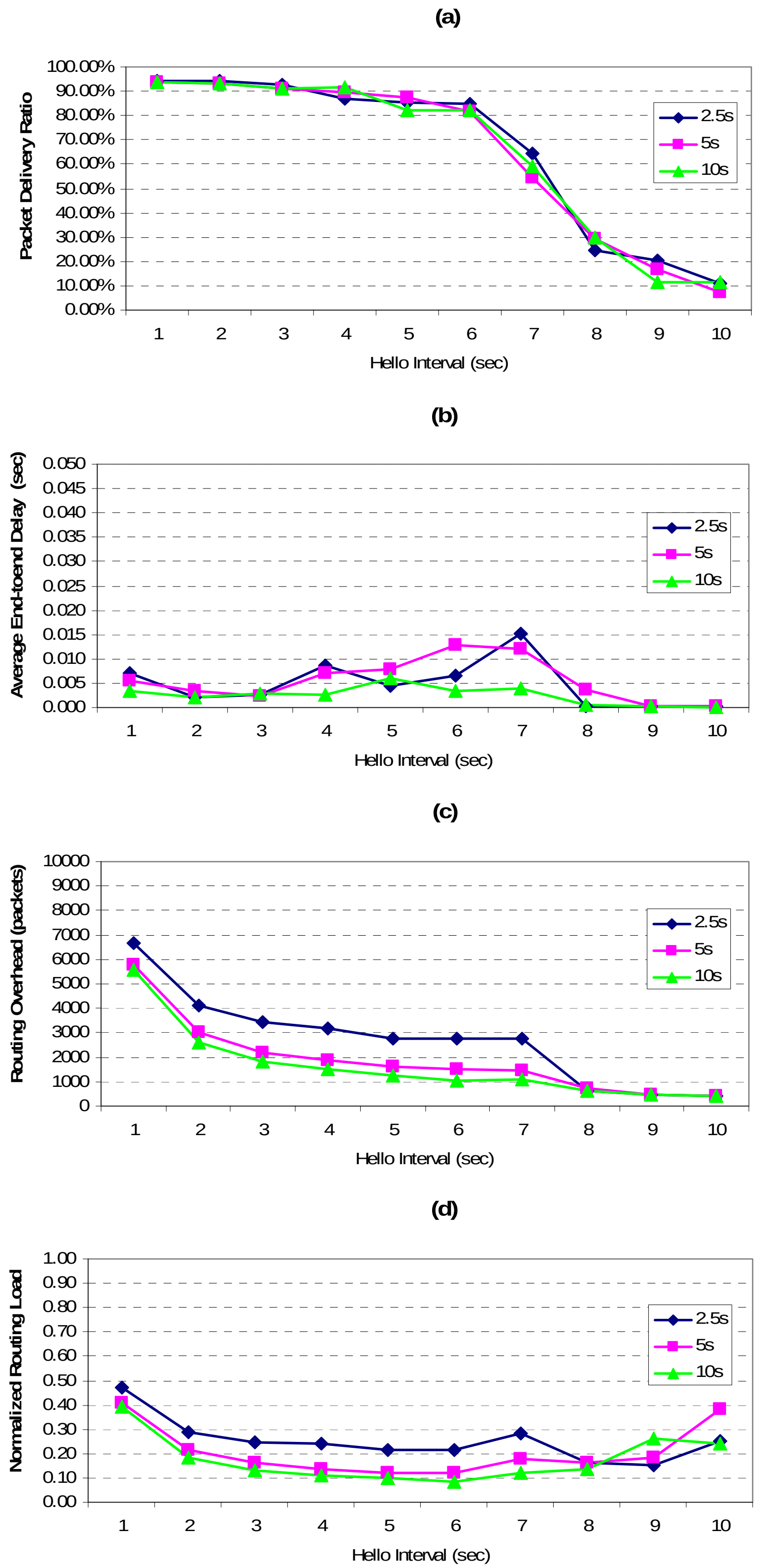


Figure 21. Performance metrics for ship network scenario by varying Hello intervals from one to 10 seconds with Topology Control intervals set at 2.5, 5, and 10 seconds.

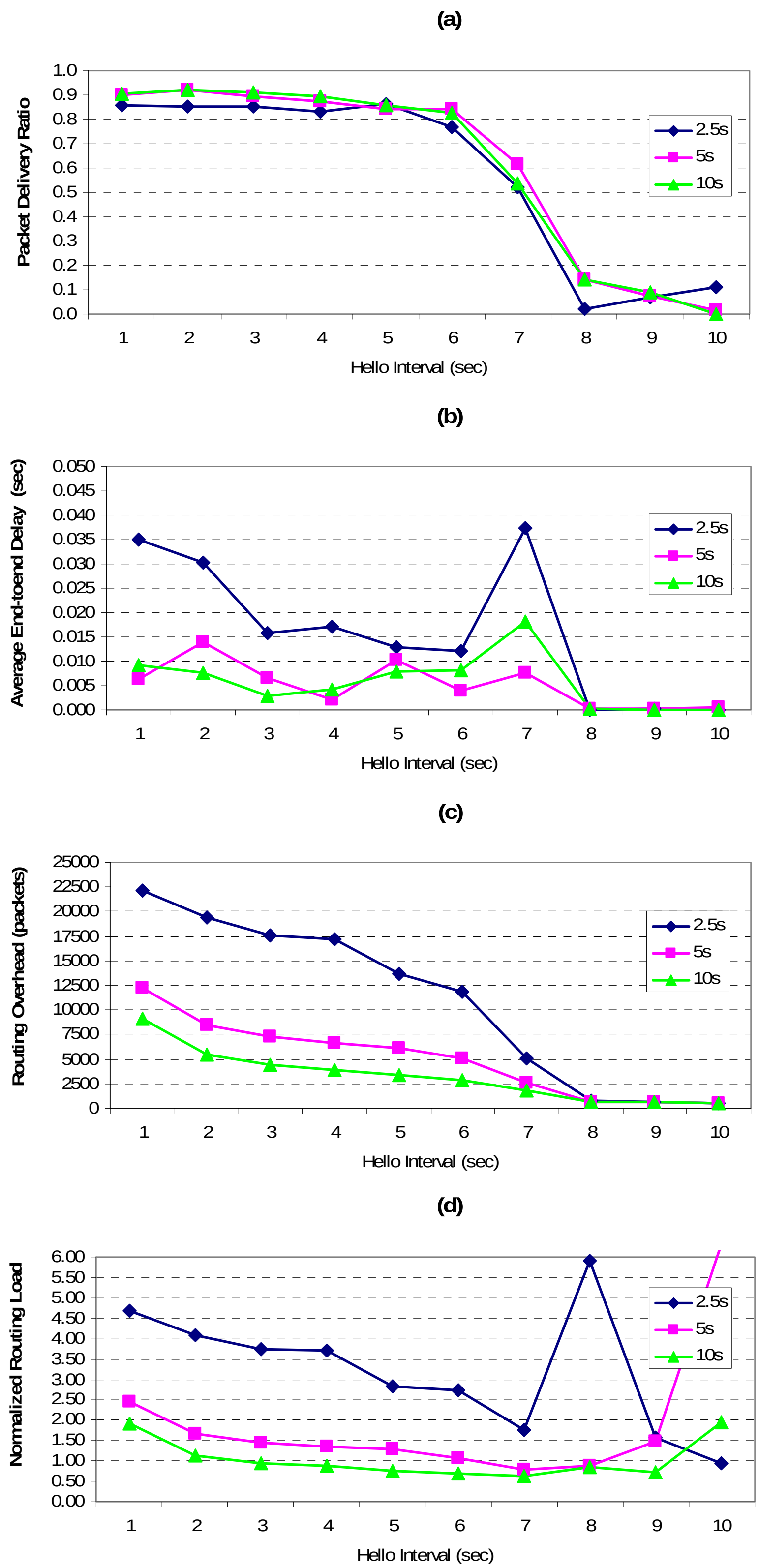


Figure 22. Performance metrics for sensor network scenario by varying Hello intervals from one to 10 seconds with Topology Control intervals set at 2.5, 5, and 10 seconds.

and is seen to complete breakdown when the Hi is >7 seconds where the PDF declined rapidly to <10%. The near-zero average end-to-end latency is simply an indication of biases to short routing paths but actually very little packets arrived at a destination. The latency performance is better for HCi of 5 and 10 seconds in the sensor network.

From Figures 21c and 21d, it can be deduced that using a Hi of between two to six seconds will result in better NRL and ROH. The shorter Hi of one second did not provide any marked improvement in PDR, i.e., the PDR is around 94% between a Hi of one and two seconds. But instead, it incurred approximately 150% more overhead packets then a Hi of two seconds. The obvious choice for Hi is between two to three seconds, as the difference in PDR and corresponding NRL is marginal. In the case of the sensor network, Figures 22c and 22d show that a TCi of 10 seconds will consistently provide better overall NRL and ROH. The results for Hi of >7 seconds showed significant instability in the network and, coupled with the non-performance in the PDR and latency metrics, will not be commented on further. Again, the range of TCi to use for the sensor network should be between two to six seconds. Table 4 provides a summary of recommended values that can be used to set up the two different scenarios defined in this thesis.

OLSR Parameters	Scenario 1 (Ship)	Scenario 2 (Sensor)
Hello Interval (Hi)	2, 3 or 4 secs	2, 3, 4 or 5 secs
Topology Control Interval (TCi)	5 or 10 secs	10 secs
Willingness (default)	3	3
Neighbor Hold Time (default)	3*TCi	3*TCi
Topology Hold Time (default)	5 secs	5 secs

Table 4. Summary of recommended parameter value for use of OLSR in the ship and sensor network scenarios.

D. RESULTS OF OLSR TUNING

Based on the recommendations provided in Table 4, a new set of simulations was conducted for the two scenarios. For ease of comparison, a Hello interval (Hi) of two seconds and topology control interval (TCi) of 10 seconds are used in both scenarios. Figures 23 to 26 and 27 to 30 provide the optimization results for the two scenarios, respectively.

1. Optimizing Ship Network Scenario

a. Influence of Mobility Rate

Figure 23a shows that by extending the TCi from 5 seconds to 10 seconds, the PDR has improved markedly and approached the performance of AODV. The improvements to average end-to-end delay are not significant. As shown in Figure 23b, the protocols generally do not exhibit problems related to latency, which are already relatively low before the optimization. The only problems resulted in increasing the TCi are in NRL and ROH. Figures 23c and 23d show that the routing loads have almost doubled. This implies that more routing packets are required to be sent to ensure route freshness. Even though the Hi is kept at two seconds, the long TCi actually resulted in more topology (TC) overhead being sent by the MPRs to discover routes. As OLSR is sensitive to node mobility, the rapid topology change can trigger an immediate transmission of a new Hello and/or TC message without waiting for the current Hello or TC transmission timer to expire. Such reactions will add more routing load to the network in order to improve the reactivity to topology change to warrant a higher PDR. In this case, the excess routing load is not a problem as it did not affect the PDR and route latency. So, the major time-sensitive applications would still work just as well in the network.

b. Influence of Node Density

Using the same arguments provided in the mobility case, the same effects can be seen when the node density is increased in a highly mobile network. Figures 24a and 24b show the same relative improvements in PDR and latency. However, this is at the expense of approximately two times the routing loads (see Figures 24c and 24d). In general, the OLSR message size grows approximately linearly as nodes are added. This increase is a result of an increase in the number of neighbors listed in the Hello message and increasing number of MPR selectors per MPR node.

c. Influence of Network Loading

Figure 25a shows the same adverse effect on the PDR when the connection rate is increased. With a long TCi, the MPRs have to react to topology changes, but in this case the node density is low and so is more likely to result in network fragmentation. Because the connection rate is high, each link breakage resulted in more dropped packets. This explains the low PDR from a connection rate of 20 pkts/sec onwards as

compared to the non-optimized case (see Figure 25b). Figures 25c and 25d show the low routing load conditions as expected because no additional MPR is likely to be introduced. Hence, a lower number of TC messages are expected to be flooded through the network.

Note that a different scale is used to allow more details to be shown in the PDR for the increasing number of connections. As can be seen from Figures 26a and 26b, the results actually reflected an optimal mix of the best both AODV and OLSR without tuning the case. The latency at >20 connections is about 52ms as compared to 37ms for the OLSR without tuning. Although the difference is marginal, the results indicated that it would not be ideal to implement too many jitter sensitive applications in this scenario. Figures 26c and 26d indicate that the routing loads are not affected by the increase in number of connections. The NRL for OLSR with TCi of 10 seconds is about two times higher than with TCi of five seconds.

2. Optimizing Sensor Network Scenario

a. Influence of Mobility Rate

In the case of sensor networking, the node density is high but mobility is low. Therefore, a slow topology change is expected which makes it ideal for the implementation of MPR flooding technique in OLSR. Figures 27a and 27b show that increasing the TCi time interval did not affect the PDR but provided marginal improvements to the metric. This is coupled with lower latency and overall routing loads. These results can be explained using the exact opposite reasons as discussed when the mobility rate is varied for the ship networking scenario. In this case, as the network topology remained largely the same, no new Hello and/or TC messages need to be transmitted until the expiration of the Hello or TC transmission timer.

b. Influence of Node Density

Figure 28a shows a similar trend but a marginally better PDR performance as compared to the earlier protocols. It is important to see that the tuning has improved the average end-to-end delay significantly. In fact, the latency is kept at <30 ms in all cases and the OLSR-tuned outperforms the earlier protocols when the node density is increased (see Figure 28b). Using the same counter arguments as provided in the ship network with high node mobility, we can see from Figures 28c and 28d that the routing overhead size grows approximately linear but more gradually as nodes are added. But

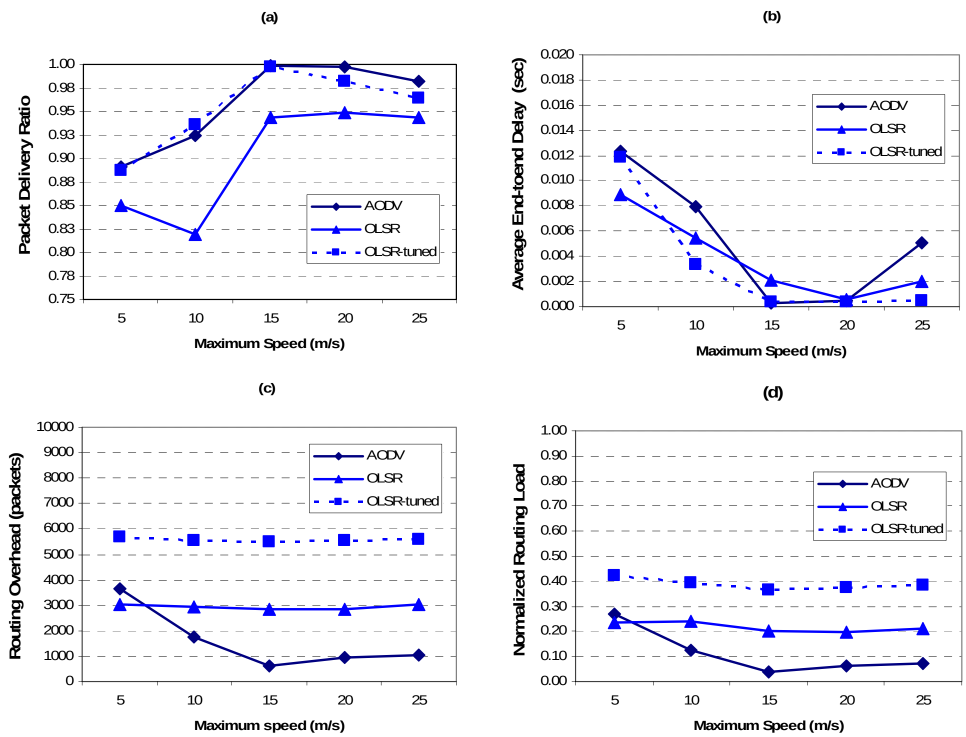


Figure 23. Results of varying mobility rate (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

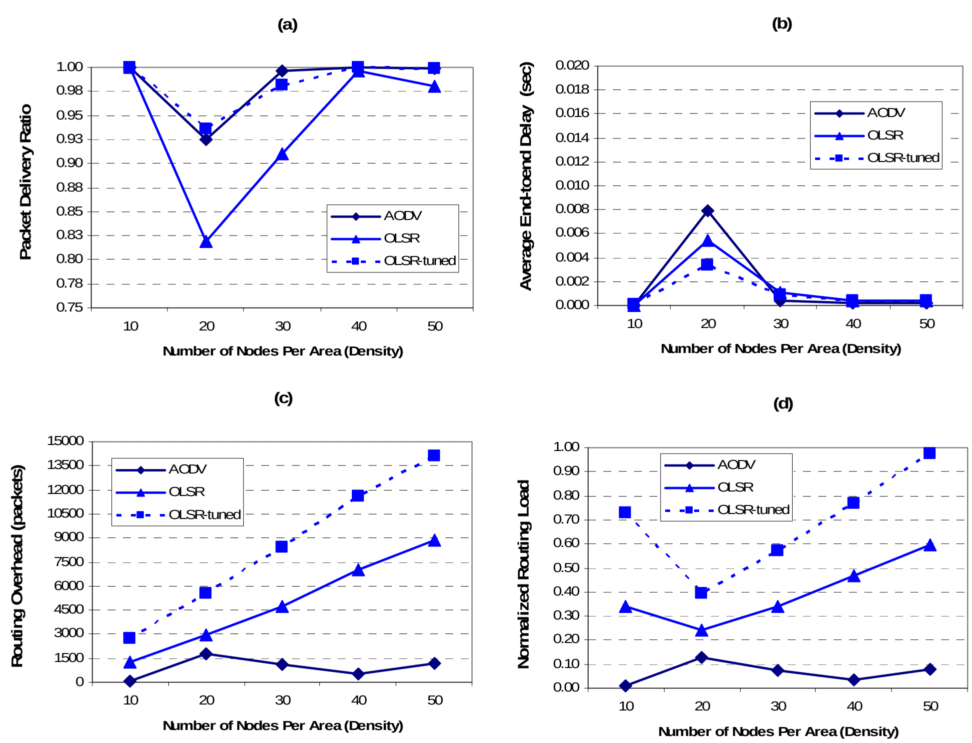


Figure 24. Results of varying node density (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

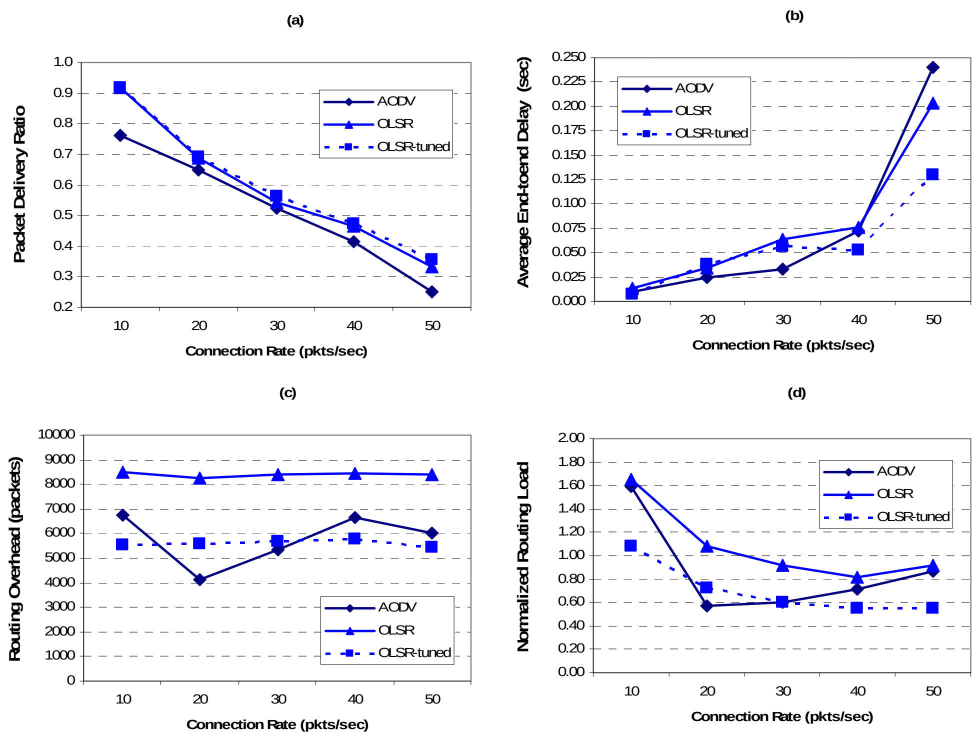


Figure 25. Results of varying connection rate (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

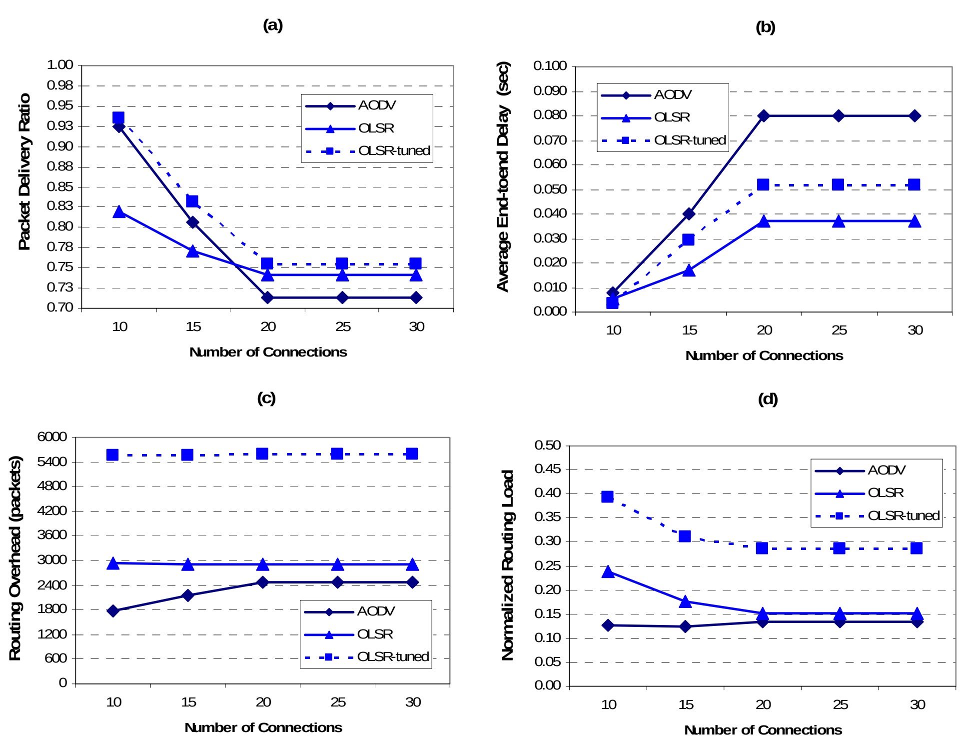


Figure 26. Results of varying number of data connections (for ship) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

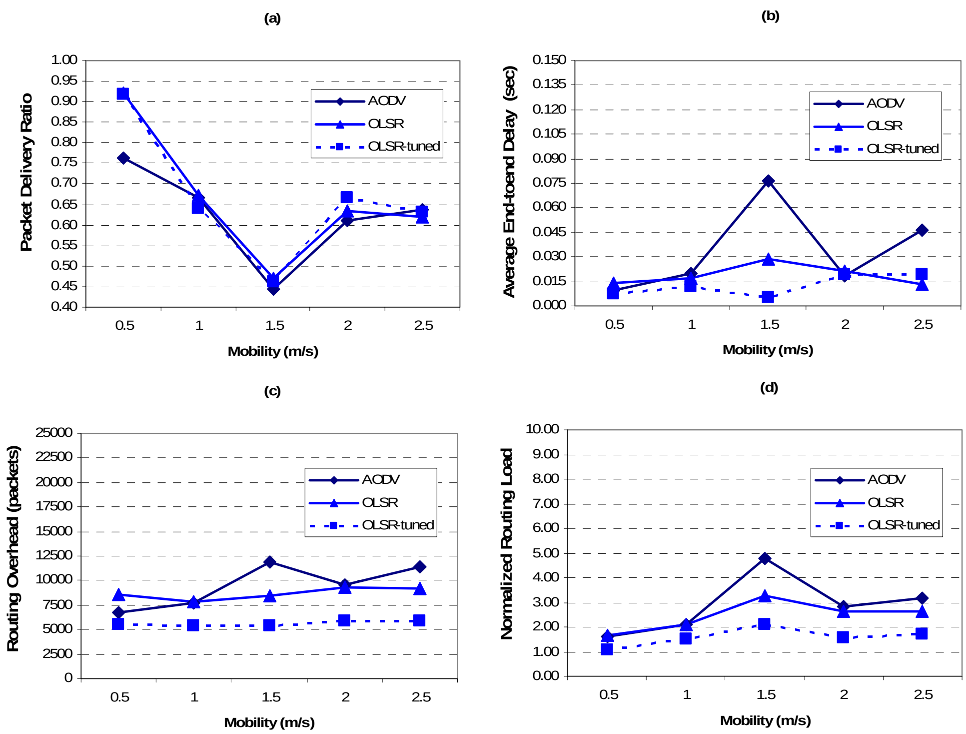


Figure 27. Results of varying mobility rate (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

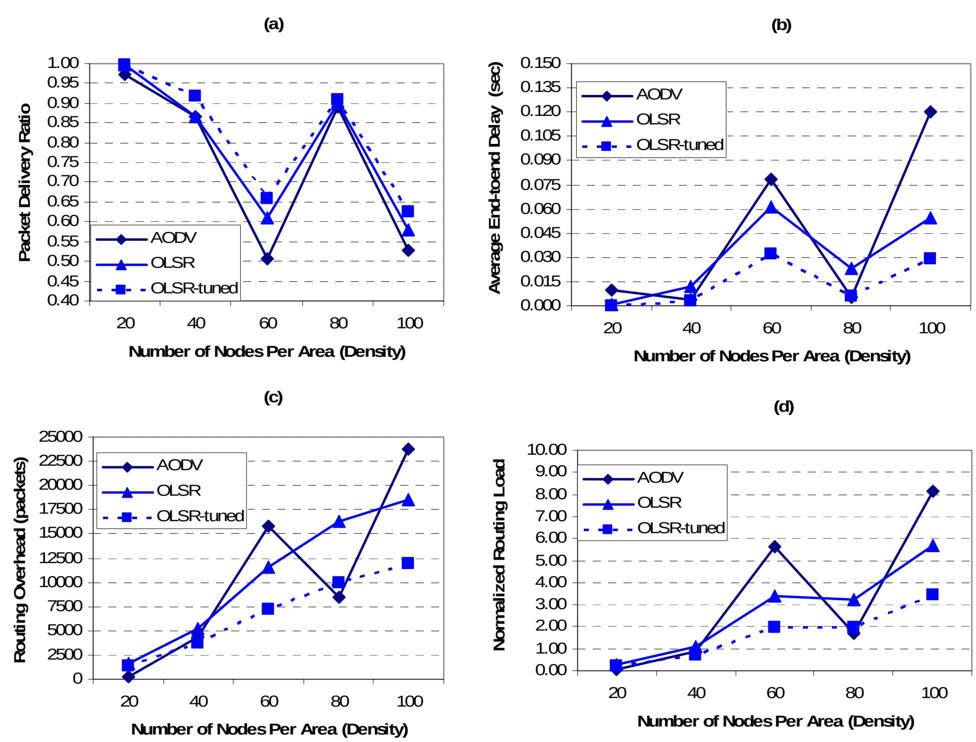


Figure 28. Results of varying node density (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

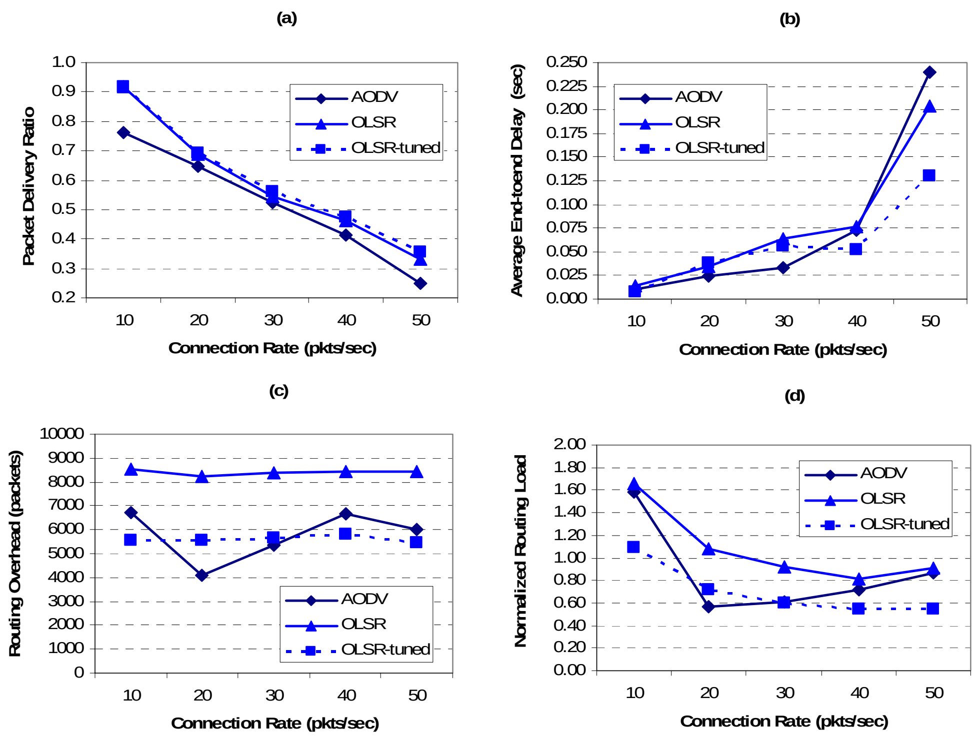


Figure 29. Results of varying connection rate (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

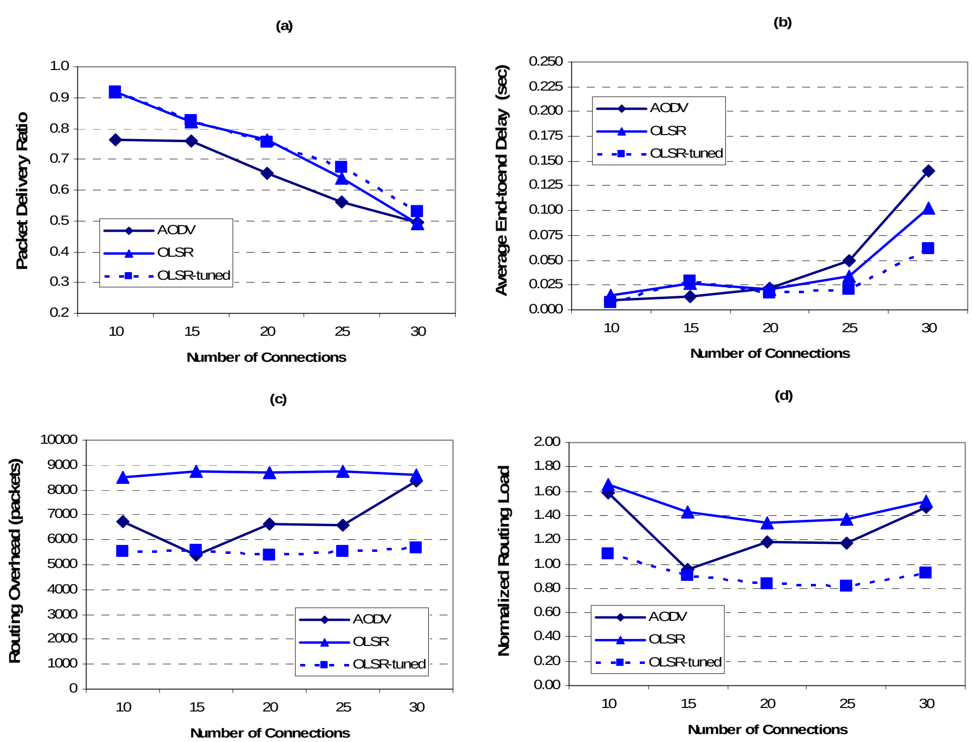


Figure 30. Results of varying number of data connections (for sensor node) on (a) PDR (b) average end-to-end delay (c) ROH and (d) NRL

the NRL for OLSR-tuned is relatively lower from 0.24 at 20 nodes to around 3.4 at 100 nodes. This compares to around 0.3 to 5.7 for the non-optimized OLSR respectively.

c. Influence of Network Loading

Figures 29 and 30 show the graphs of varying the network load. From the results of the performance metrics, it can generally be concluded that the optimization of OLSR has shown to consistently outperform the default implementation of OLSR as well as AODV under all network loading conditions.

E. SUMMARY

In this chapter, the simulations for the tactical networks for ships and sensor-based network scenarios are conducted. The results of the mobility, node density and network loading tests based on the performance metrics are discussed. In general, the classic AODV routing protocol has shown to perform relatively better than DSDV and OLSR for the tactical ship networks. For sensor networks, OLSR has outperformed both AODV and DSDV in most metrics for that unique networking environment. Lastly, the default values of Hello and topology control intervals for OLSR are tweaked and the optimal performance values proposed. The simulation results after the tuning showed that improvements can be made to further enhance the performance of OLSR.

The next chapter concludes the analysis and the findings. Future possible research areas are also discussed.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS, RECOMMENDATIONS, AND FUTURE WORK

A. CONCLUSIONS

This thesis provides a performance analysis of three different mobile ad-hoc routing protocols, namely, DSDV, AODV and OLSR by means of simulation using an open-source network simulator software called NS-2. The details at different aspects that affect the scalability and interoperability of the routing protocols over tactical naval ship network operations and sensor base networks are analyzed. This is achieved through utilizing the array of tools that is available in NS-2 for simulating an accurate mobile node model. The data traffic and group mobility models are also carefully chosen and configured to closely simulate the operational environment of both scenarios. The behaviors of the routing protocols are tested based on the influence of node mobility rate, density, and network loads and measured using various performance metrics. In the second part of the thesis, the default parameters of OLSR based on RFC3625 are varied by tuning the Hello (Hi) and Topology Control (TCi) intervals that can influence the network performance.

In an ad-hoc network environment where the mobile nodes are expected to inherit high group mobility such as the tactical ship networks, the classic AODV remains the protocol of choice, as opposed to OLSR and DSDV. Although OLSR is a vast improvement over the older proactive or table-driven DSDV protocol, it did not perform as well as AODV in an open and dynamic networking environment akin to operations of that in a littoral theatre. When group mobility is accounted for, in a sparse network where node density is in the order of 20 to 30 ships, AODV is less affected by network fragmentation. Additionally, it is able to sustain a superior PDR of >90% in all cases while keeping the route latency to within 15ms. This is important for supporting delay-sensitive applications such as digitized voice and video. However, in a situation where the network traffic is heavy, approaching 60% of the overall network throughput, it is necessary to consider putting some QoS protection strategies in a contention based network such as IEEE 802.11. All of the protocols are seen to be affected severely in this situation. The poor performance of OLSR is caused primarily by the relatively high administrative routing overhead, which although not apparent in the observed PDR and average end-to-end de-

lay, could become a challenging issue in actual implementation if not appropriately controlled. This is because if we add the real-world problems associated with propagation channels, any excessive overhead packets would only increase the chances of packet collision and cause more latency issues. While this is unlikely to be a problem for best-effort traffic, it can affect delay-sensitive applications such as digitized voice and video.

On the other hand, OLSR has stood the tests and acclaimed accolades to emerge as the best protocol to use in the sensor network based scenario. It has consistently outperformed AODV and most definitely DSDV in all cases. Although the poor ROH of OLSR has continued to be a concern, in this case, the exception is that even AODV would have similar ROH problems due to the high node density and low node mobility network environment. AODV failed to perform as well in this scenario especially when the node density is increased. This is due in part to its poor routing strategy in a network that has relatively static topology. It also scaled poorly when the density is in the region of 100 nodes or more. While the choice of protocol is clear in the sensor network scenario, it necessitates a need to balance between higher bandwidth demands and low latency. It is shown that for nominal node parameters when either the connection rate is 40 pkts/sec or number of connections is 25 and higher, the overall latency will increase rapidly as to hinder the use of delay-sensitive applications. Best effort traffic is typically not a problem in all cases.

The results of the OLSR tuning have shown that it is possible to achieve optimality in the routing protocol performance simply by varying the Hi and TCi. While it can improve the protocol reactivity to link failures, there is generally no impact on the bandwidth performance. The tuning resulted in better optimization of the OLSR for the sensor networks than for the ship networks. The improvements are significant and seem to apply to all the tests and performance metrics in the sensor network scenario. Whereas for the ship networks, the marginal improvements made to the PDR and latency is actually offset by the increasing routing overheads in all cases.

B. RECOMMENDATIONS

The results provided by the simulations are certainly not an end by themselves but by constantly improving all possible aspects of the simulation environment will only provide better confidence in their use, including the transition to experimentation and real

implementations. Certainly, due to the constraints of time and potential complexity, there are some limitations that are assumed for the simulations and modeling of the mobile networking attributes. The following will highlight some key areas that may immediately be addressed without involving too drastic a change in the basic NS-2 code or recompilations, but can provide deeper insight as to why the protocols performed the way they did in the given scenarios. This is especially important for cases where there are obvious divergences from the given theories. More abstract approaches are provided under future research areas. The more immediate areas that can be addressed include, but are not limited to, the following:

- Adding other metrics that would reveal the problems of poor protocol performance in addition to capturing the performance of a protocol. To this end, the authors in [48] suggest reviewing the reasons behind any poor PDR by inspecting where and why the packets are dropped.
- Checking the reasons behind the high routing overhead, especially in the cases where the PDR is observed to decline rapidly with increasing traffic load. This requires looking into the trace file to sieve out information pertaining to:

  (a) how many neighbors successfully received a broadcast RREQ;
  (b) what was the percentage of successful RREP transmissions;
  (c) what was the percentage of successful data packet transmissions;
- Accounting for the MAC load to find out the number of routing, Address Resolution Protocol (ARP), and control (e.g., RTS, CTS, ACK) packets transmitted by the MAC layer for each delivered data packet. The MAC load is a measure of effective utilization of the wireless medium by data traffic. The granular details will provide better insight as to what contributes to the high routing loads.
- Adding an energy model into the OTcl script to investigate the energy consumption of the mobile nodes. This finding will provide another dimension to measure the overall performance characteristics of OLSR, especially in its use over sensor networks.
- Scheduling and queue management. Instead of using the simple priority queue model with tail drop, other techniques such as Weighted Fair Queuing (WFQ), Random Early Drop (RED) and other variants, should be tested as a means of QoS control.
- Consider using other military related configurations to model the physical layer and review the routing protocol performance. For example, real-world military environments are likely to deviate from the standard commercial 802.11 WLAN specifications. There exist other variants of

802.11 based wireless solutions that use different frequency spectra, transmit power, receiver sensitivity, and data rates.

- Improving the sensor node modeling. The research interest in this area is evidently very immense and there is much contributed code that can be found in the NS-2 website which provide MANET extension tools that support sensor networking.

C. FUTURE RESEARCH AREAS

The open research problems for MANET are all encompassing. In this section a short list of some of the key areas in network simulation is provided. This can be used in future thesis research work to either augment the findings made in this thesis or be conducted independently.

1. Cross Layering Issues

There are recent efforts to improve the performance of MANET in an environment of random variations in the physical channel and network connectivity. The modeling requires using a management sublayer to handle the interactions between different layers in the protocol stack. Using this approach called cross layering, resource allocation and decisions will be done with a more global view of the network and can reduce instabilities of the wireless network. Some related work can be referenced in [49] and [50].

2. MAC Layer Adaptations

Other than the IEEE 802.11 DCF MAC layer implementation, it would be useful to include simulations that can adapt to the IEEE 802.11 Point Coordinated Function (PCF) as a means of traffic classification and prioritization. The idea is to allow a station to become a Point Coordinator (PC) and let that node send beacons, initiate Contention Free Periods (CFPs), and poll other stations during these CFPs in order to give higher priorities to such stations. This could result in good traffic provisioning for higher priority and delay-sensitive applications.

3. Security

In the context of military operations, the importance of security in wireless communications is never overstated. An emphasis on information security means that it is not sufficient that just the routing part works. It requires that the information be exchanged in a secure and timely manner as well. This is a challenge that any implementation of new protocols must take into consideration. Reference in [20] provides extensive

research into the security protocols for ad-hoc networks. There are extensions for NS-2 that can be found on the website that would support some of them, namely, Ariadne and Secure Link State Protocol (SEAD). While the advantages of including security mechanisms are clear, the impact on the general network performance needs to be assessed. Any simulation research work in these areas has to consider the software support issues of any 3rd party contributed code unless they are independently developed.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. SAMPLE TCL SCRIPT FILE

This is the working simulation script for configuring the mobile node model. Depending on the scenarios, the traffic and mobility are loaded accordingly and the parameters are varied for each run.

```
# ======================================================================
# MANET simulation script using OLSR as Routing Protocol
# Author : G L Pore, gpore@nps.edu
#======================================================================
#======================================================================
# Define options
#======================================================================
set opt(chan) Channel/WirelessChannel
# set opt(prop) Propagation/Ricean  ;# only useful for ns-2.1b7 and be-
low
set opt(prop) Propagation/TwoRayGround
set opt(netif) Phy/WirelessPhy
set opt(mac) Mac/802_11
set opt(ifq) Queue/DropTail/PriQueue
set opt(ll) LL
set opt(ant) Antenna/OmniAntenna
set opt(x) 200          ;# X dimension of the topography
set opt(y) 200          ;# Y dimension of the topography
set opt(ifqlen) 50      ;#packet in ifq
set opt(tr) final_sources1.tr       ;# trace filename
set opt(nam) olsr-n50M05xy200r10.nam      ;# nam trace file
set opt(adhocRouting) OLSR          ;#routing protocol
set opt(nn) 50          ;# how many nodes are simulated
set opt(cp) "cbr-n50-mc10r10s1" ;# Traffic generation file
set opt(sc) "RPGM-n50h10xy200a50r50.ns_movements" ;# Mobility File
set opt(stop) 100.0     ;# simulation time
set opt(seed) 0.0 ;#seed for Random Number Generator

# ======================================================================
# Check for random seed
# ======================================================================

if {$opt(seed) > 0} {
    puts "Seeding Random number generator with $opt(seed)\n"
    ns-random $opt(seed)
}

# ======================================================================
# Other default settings
# ======================================================================
LL set mindelay_ 50us
LL set delay_ 25us
Agent/TCP set packetSize_ 512
Queue/DropTail/PriQueue set Prefer_Routing_Protocols 1
#---------------------------------------------------------------------
# unity gain, omni-directional antennas at height of 1.5 meters
#---------------------------------------------------------------------
Antenna/OmniAntenna set X_ 0
```

```
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 0.1
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0
#---------------------------------------------------------------------
# Initialize the SharedMedia interface with parameters to make
# it work like the 914MHz Lucent WaveLAN DSSS radio interface
#---------------------------------------------------------------------
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.2818
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0
# ====================================================================
# Main Program
# ====================================================================
#=====================================================================
# Initialize Global Variables
#=====================================================================
# create simulator instance
set ns_ [new Simulator]
#=====================================================================
# set wireless channel, radio-model and topography objects
#=====================================================================
#set wchan [new $opt(chan)]
#set wprop [new $opt(prop)]
set wtopo [new Topography]
#=====================================================================
# create trace object for ns and nam
#=====================================================================
set tracefd [open $opt(tr) w]
set namtrace [open $opt(nam) w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)
#=====================================================================
# use new trace file format
#=====================================================================
$ns_ use-newtrace
#=====================================================================
# define topology
#=====================================================================
$wtopo load_flatgrid $opt(x) $opt(y)
#=====================================================================
# Create God
#=====================================================================
set god_ [create-god $opt(nn)]
#=====================================================================
# global node setting - defines how mobile node should be created
#=====================================================================
$ns_ node-config -adhocRouting $opt(adhocRouting) \
-llType $opt(ll) \
-macType $opt(mac) \
-ifqType $opt(ifq) \
-ifqLen $opt(ifqlen) \
-antType $opt(ant) \
```

```
-propType $opt(prop) \
-phyType $opt(netif) \
-channelType $opt(chan) \
-topoInstance $wtopo \
-agentTrace ON \
-routerTrace ON \
-macTrace ON \
-MovementTrace OFF

#=====================================================================
# Specifics to UM-OLSR - Here to specify the parameters for OLSR
#=====================================================================
#
# control OLSR behaviour from this script -
# commented lines are not needed because
# those are default values
#
Agent/OLSR set use_mac_    true
#Agent/OLSR set debug_      false
#Agent/OLSR set willingness 3
#Agent/OLSR set hello_ival_ 2
#Agent/OLSR set tc_ival_    5

#======================================================================
# Only if Ricean Fading model is used, these are the parameters to be
# uncommented for mobile node logging
#======================================================================
# set prop_inst [$ns_ set propInstance_]
# $prop_inst MaxVelocity  2.5;
# $prop_inst RiceanK        6;
# $prop_inst LoadRiceFile  "rice_table.txt";

#======================================================================
# Create the specified number of nodes [$opt(nn)] and “attach” them
# to the channel.
#======================================================================
for {set i 0} {$i < $opt(nn) } {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 ;# disable random motion
}

#======================================================================
# Define node movement model
#======================================================================
puts "Loading connection pattern..."
source $opt(cp)
#======================================================================
# Define traffic model
#======================================================================
puts "Loading scenario file..."
source $opt(sc)
# Define node initial position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
# 35 defines the node size in nam, adjust it according to your scenario
# The function must be called after mobility model is defined
$ns_ initial_node_pos $node_($i) 35
}
```

```
#=====================================================================
# Tell nodes when the simulation ends
#=====================================================================
for {set i 0} {$i < $opt(nn) } {incr i} {
$ns_ at $opt(stop).000000001 "$node_($i) reset";
}
#=====================================================================
# tell nam the simulation stop time
#=====================================================================
$ns_ at $opt(stop) "$ns_ nam-end-wireless $opt(stop)"
$ns_ at $opt(stop).000000001 "puts \"NS EXITING...\" ; $ns_ halt"
puts "Starting Simulation..."
$ns_ run
#=====================================================================
# END OF SIMULATION SCRIPT
#=====================================================================
```

APPENDIX B. SAMPLE MOBILITY OUTPUT FILES GENERATED BY BONNMOTION SOFTWARE

```
#======================================================================
# RPGM configuration parameters in ".param" file
#======================================================================

model=RPGM
ignore=1000.0
randomSeed=1138684071284
x=200.0
y=200.0
duration=100.0
nn=50
circular=false
groupsize_E=50.0
groupsize_S=2.0
pGroupChange=0.0
maxdist=50.0
minspeed=0.5
maxspeed=0.5
maxpause=100.0

#======================================================================
# A snapshot of the mobility file in ".ns_movement" file
#======================================================================

$node_(0) set X_ 81.58262414851012
$node_(0) set Y_ 130.6066071706863
$ns_ at 0.0 "$node_(0) setdest 104.90280431890744 128.45480832524777
0.5000000000000001"
$ns_ at 46.838490216958235 "$node_(0) setdest 124.20255797191567
111.16345086228074 0.4874347603173074"
$node_(1) set X_ 91.44788112690092
$node_(1) set Y_ 120.48913548750758
$ns_ at 0.0 "$node_(1) setdest 104.23300177401077 140.11059648025397
0.49999999999999983"
$ns_ at 46.838490216958235 "$node_(1) setdest 114.97308526618576
126.84195260664286 0.3211087682829759"
$node_(2) set X_ 86.70346581657529
$node_(2) set Y_ 120.91594706878503
$ns_ at 0.0 "$node_(2) setdest 106.94776865402164 132.69003913895267
0.5"
$ns_ at 46.838490216958235 "$node_(2) setdest 114.784399370952
128.2213978808966 0.16969366675245842"
$node_(3) set X_ 112.12790308891168
$node_(3) set Y_ 135.96002953667332
$ns_ at 0.0 "$node_(3) setdest 102.38497401413859 114.66363607451721
0.5000000000000004"
$ns_ at 46.838490216958235 "$node_(3) setdest 112.96160834238071
98.09115923876658 0.36981479940978024"
$node_(4) set X_ 84.41948199455476
$node_(4) set Y_ 108.78410001097805
```

```
$ns_ at 0.0 "$node_(4) setdest 94.47301379717668 129.93563753770863
0.4999999999999996"
$ns_ at 46.838490216958235 "$node_(4) setdest 102.2583692827101
115.63153416335419 0.3063409955810473"
$node_(5) set X_ 96.6474098012414
$node_(5) set Y_ 126.28420222369269
$ns_ at 0.0 "$node_(5) setdest 115.23163244809393 140.53508675986222
0.49999999999999983"
$ns_ at 46.838490216958235 "$node_(5) setdest 129.10385024803998
119.88361406769371 0.4679727516308025"
$node_(6) set X_ 85.41175070637895
$node_(6) set Y_ 137.86790969773205
$ns_ at 0.0 "$node_(6) setdest 88.49163476796356 161.08375246981808
0.4999999999999996"
$ns_ at 46.838490216958235 "$node_(6) setdest 107.19870859901766
146.1342408645707 0.4504510829570569"
$node_(7) set X_ 111.99486800281402
$node_(7) set Y_ 132.34671319405862
$ns_ at 0.0 "$node_(7) setdest 112.01111225229597 148.4476015847156
0.3437535349777135"
$ns_ at 46.838490216958235 "$node_(7) setdest 116.62836850305209
139.27810464674272 0.19311694445403588"
$node_(8) set X_ 107.49890051273042
$node_(8) set Y_ 133.6128188855768
$ns_ at 0.0 "$node_(8) setdest 111.84989377044622 130.13583796226328
0.11891093113005725"
$ns_ at 46.838490216958235 "$node_(8) setdest 118.47147434726045
118.6116868378744 0.25001221125876144"
$node_(9) set X_ 104.80651459806123
$node_(9) set Y_ 111.03382762302866
$ns_ at 0.0 "$node_(9) setdest 99.08867100821578 133.74433854051102
0.4999999999999997"
$ns_ at 46.838490216958235 "$node_(9) setdest 107.44076102870693
116.70746425865772 0.3569122708366572"
$node_(10) set X_ 89.63962089406981
$node_(10) set Y_ 136.37478640393937
$ns_ at 0.0 "$node_(10) setdest 100.25119750588641 115.4976342211686
0.5000000000000001"
$ns_ at 46.838490216958235 "$node_(10) setdest 110.56233071730587
99.87614878431057 0.35209019028845134"
$node_(11) set X_ 76.12917922569889
$node_(11) set Y_ 117.13403080327579
$ns_ at 0.0 "$node_(11) setdest 89.23567246374668 136.5422990529685
0.4999999999999998"
$ns_ at 46.838490216958235 "$node_(11) setdest 113.33552304586993
127.18659444448176 0.48629390208619516"
$node_(12) set X_ 88.30434299396732
$node_(12) set Y_ 106.03446200307282
$ns_ at 0.0 "$node_(12) setdest 103.72761791059264 123.6578451146442
0.5000000000000001"
$ns_ at 46.838490216958235 "$node_(12) setdest 123.0680297213186
108.12374189077784 0.46662423053610136"
$node_(13) set X_ 68.85668712765752
$node_(13) set Y_ 123.00572725482087
$ns_ at 0.0 "$node_(13) setdest 76.60343861946647 145.106606938109
0.49999999999999956"
```

APPENDIX C. SAMPLE TRAFFIC FILE

```
#=======================================================================
# A snapshot of traffic file generated using cbrgen.tcl in NS-2
#=======================================================================
#
#=======================================================================
# nodes: 50, max conn: 10, send rate: 0.10000000000000001, seed: 1
#=======================================================================
#
# 1 connecting to 2 at time 2.5568388786897245
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.10000000000000001
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 "$cbr_(0) start"
#
# 4 connecting to 5 at time 56.333118917575632
#
set udp_(1) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(1)
set null_(1) [new Agent/Null]
$ns_ attach-agent $node_(5) $null_(1)
set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) set packetSize_ 512
$cbr_(1) set interval_ 0.10000000000000001
$cbr_(1) set random_ 1
$cbr_(1) set maxpkts_ 10000
$cbr_(1) attach-agent $udp_(1)
$ns_ connect $udp_(1) $null_(1)
$ns_ at 56.333118917575632 "$cbr_(1) start"
#
# 4 connecting to 6 at time 146.96568928983328
#
set udp_(2) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(2)
set null_(2) [new Agent/Null]
$ns_ attach-agent $node_(6) $null_(2)
set cbr_(2) [new Application/Traffic/CBR]
$cbr_(2) set packetSize_ 512
$cbr_(2) set interval_ 0.10000000000000001
$cbr_(2) set random_ 1
$cbr_(2) set maxpkts_ 10000
$cbr_(2) attach-agent $udp_(2)
$ns_ connect $udp_(2) $null_(2)
$ns_ at 146.96568928983328 "$cbr_(2) start"
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. SAMPLE TRACE FILE

```
#======================================================================
# A snapshot of output trace file showing node positions at zero
# second and traffic information up to 0.001107283s
#======================================================================
#
M 0.00000 0 (133.56, 101.39, 0.00), (153.46, 130.78), 0.60
M 0.00000 1 (135.92, 141.48, 0.00), (184.21, 142.04), 0.82
M 0.00000 2 (150.40, 146.96, 0.00), (166.70, 90.58), 1.00
M 0.00000 3 (165.56, 96.93, 0.00), (183.22, 122.46), 0.77
M 0.00000 4 (143.45, 142.48, 0.00), (165.39, 154.86), 0.43
M 0.00000 5 (166.94, 94.68, 0.00), (155.07, 132.51), 0.67
M 0.00000 6 (132.54, 97.72, 0.00), (116.06, 110.90), 0.36
M 0.00000 7 (154.35, 122.40, 0.00), (116.33, 132.33), 0.67
M 0.00000 8 (159.67, 92.74, 0.00), (153.16, 142.88), 0.86
M 0.00000 9 (166.80, 97.62, 0.00), (163.65, 140.07), 0.72
M 0.00000 10 (189.08, 110.24, 0.00), (196.99, 158.98), 0.84
M 0.00000 11 (124.94, 99.71, 0.00), (146.97, 137.22), 0.74
M 0.00000 12 (156.36, 124.35, 0.00), (118.15, 117.92), 0.66
M 0.00000 13 (179.98, 119.90, 0.00), (183.75, 131.62), 0.45
M 0.00000 14 (152.68, 129.93, 0.00), (113.81, 116.30), 0.70
M 0.00000 15 (141.13, 135.64, 0.00), (156.63, 145.29), 0.31
M 0.00000 16 (156.18, 133.06, 0.00), (159.15, 148.03), 0.26
M 0.00000 17 (155.99, 166.80, 0.00), (137.43, 124.68), 0.78
M 0.00000 18 (162.01, 120.93, 0.00), (178.39, 112.41), 0.31
M 0.00000 19 (129.66, 122.86, 0.00), (155.92, 123.92), 0.45
M 0.00000 20 (116.55, 113.53, 0.00), (156.74, 132.04), 0.75
M 0.00000 21 (174.22, 112.42, 0.00), (189.41, 125.16), 0.34
M 0.00000 22 (180.37, 135.14, 0.00), (162.17, 125.65), 0.35
M 0.00000 23 (155.69, 108.12, 0.00), (161.06, 138.60), 0.53
M 0.00000 24 (159.15, 110.16, 0.00), (146.80, 133.62), 0.45
M 0.00000 25 (163.21, 134.92, 0.00), (137.01, 130.05), 0.45
M 0.00000 26 (137.51, 157.67, 0.00), (143.88, 119.93), 0.65
M 0.00000 27 (152.69, 168.07, 0.00), (150.82, 161.69), 0.11
M 0.00000 28 (137.11, 136.65, 0.00), (179.32, 127.29), 0.74
M 0.00000 29 (172.44, 154.95, 0.00), (154.82, 108.39), 0.85
M 0.00000 30 (160.62, 138.10, 0.00), (176.27, 119.95), 0.41
M 0.00000 31 (158.25, 168.64, 0.00), (141.94, 139.68), 0.57
M 0.00000 32 (172.08, 104.48, 0.00), (151.32, 127.07), 0.52
M 0.00000 33 (145.35, 129.47, 0.00), (120.68, 97.98), 0.68
M 0.00000 34 (156.43, 129.76, 0.00), (135.31, 117.88), 0.41
M 0.00000 35 (122.09, 105.39, 0.00), (162.04, 112.23), 0.69
M 0.00000 36 (127.78, 141.50, 0.00), (165.79, 105.62), 0.89
M 0.00000 37 (140.44, 119.06, 0.00), (161.61, 114.05), 0.37
M 0.00000 38 (159.37, 118.26, 0.00), (162.56, 91.71), 0.45
M 0.00000 39 (135.20, 139.50, 0.00), (145.94, 125.71), 0.30
M 0.00000 40 (168.67, 126.73, 0.00), (160.16, 154.88), 0.50
M 0.00000 41 (145.25, 130.25, 0.00), (152.89, 128.73), 0.13
M 0.00000 42 (152.57, 91.47, 0.00), (151.88, 130.04), 0.66
M 0.00000 43 (156.45, 105.80, 0.00), (160.36, 125.55), 0.34
M 0.00000 44 (126.19, 105.40, 0.00), (150.25, 157.17), 0.97
M 0.00000 45 (148.41, 120.95, 0.00), (133.06, 139.20), 0.41
M 0.00000 46 (158.51, 131.66, 0.00), (155.38, 109.07), 0.39
M 0.00000 47 (121.98, 141.22, 0.00), (168.23, 111.52), 0.94
```

```
M 0.00000 48 (141.99, 120.63, 0.00), (112.23, 139.24), 0.60
M 0.00000 49 (124.40, 107.69, 0.00), (151.21, 150.10), 0.85
s -t 0.000087269 -Hs 30 -Hd -1 -Ni 30 -Nx 160.62 -Ny 138.10 -Nz 0.00 -
Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 30.255 -Id -
1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps 0 [-Pt HELLO
-Po 30 -Ph 0 -Pms 0]
s -t 0.000169789 -Hs 0 -Hd -1 -Ni 0 -Nx 133.56 -Ny 101.39 -Nz 0.00 -Ne
-1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.255 -Id -1.255
-It OLSR -Il 48 -If 0 -Ii 1 -Iv 32 -P olsr -Pn 1 -Ps 0 [-Pt HELLO -Po 0
-Ph 0 -Pms 0]
s -t 0.000282269 -Hs 30 -Hd -1 -Ni 30 -Nx 160.62 -Ny 138.10 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 100 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
s -t 0.000604789 -Hs 0 -Hd -1 -Ni 0 -Nx 133.56 -Ny 101.39 -Nz 0.00 -Ne
-1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 0 -Mt 800 -Is 0.255 -
Id -1.255 -It OLSR -Il 100 -If 0 -Ii 1 -Iv 32 -P olsr -Pn 1 -Ps 0 [-Pt
HELLO -Po 0 -Ph 0 -Pms 0]
r -t 0.001082283 -Hs 25 -Hd -1 -Ni 25 -Nx 163.21 -Ny 134.92 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082292 -Hs 16 -Hd -1 -Ni 16 -Nx 156.18 -Ny 133.06 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082292 -Hs 46 -Hd -1 -Ni 46 -Nx 158.51 -Ny 131.66 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082301 -Hs 34 -Hd -1 -Ni 34 -Nx 156.43 -Ny 129.76 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082307 -Hs 14 -Hd -1 -Ni 14 -Nx 152.68 -Ny 129.93 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082315 -Hs 2 -Hd -1 -Ni 2 -Nx 150.40 -Ny 146.96 -Nz 0.00 -Ne
-1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is 30.255
-Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps 0 [-Pt
HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082316 -Hs 40 -Hd -1 -Ni 40 -Nx 168.66 -Ny 126.74 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001082317 -Hs 12 -Hd -1 -Ni 12 -Nx 156.36 -Ny 124.35 -Nz 0.00 -
Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
r -t 0.001107283 -Hs 25 -Hd -1 -Ni 25 -Nx 163.21 -Ny 134.92 -Nz 0.00 -
Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md ffffffff -Ms 1e -Mt 800 -Is
30.255 -Id -1.255 -It OLSR -Il 48 -If 0 -Ii 0 -Iv 32 -P olsr -Pn 1 -Ps
0 [-Pt HELLO -Po 30 -Ph 0 -Pms 0]
```

APPENDIX E. JAVA PARSER CODE

This program is adapted from the codes provided by K. Sadasivam in [37]. It is used for parsing the trace files to derive the packet delivery ratio, average end-to-end delay, routing overhead packets, and normalized routing load.

```
#======================================================================
# Java parser codes for deriving performance metrics
#======================================================================
import java.util.*;
import java.lang.*;
import java.io.*;
public class parsetrace  {

  public static void main (String args[]) {
   String s, thisLine, currLine,thisLine1;
   int j=0;
   FileInputStream  fin,fin1;
   FileOutputStream fout,fout1;
   final int FILES = 45;
   final int MAX_PACKETS = 400000;

  try {
       int i=0, sends=0, receives=0;
       int drops=0,packet_id=0, highest_packet_id = 0;
       int line_count=0,current_line=0, routing_packets=0;
       int count=0;

       float pdfraction, time=0, packet_duration=0, end_to_end_delay=0;
       float avg_end_to_end_delay=0;
       float start_time[] = new float[MAX_PACKETS];
       float end_time[] = new float[MAX_PACKETS];
       float sent_packets[] = new float[MAX_PACKETS];
       float received_packets[] = new float[MAX_PACKETS];
       String tokens[] = new String[100];

       // initialize the start time
       for (i=0; i<MAX_PACKETS ; i++)
       start_time[i] = 0;

       fout =new FileOutputStream ("traceoutput.txt");
       DataOutputStream op = new DataOutputStream(fout);

       for (int h=0;h<FILES+1;h++) {
         j=0;
         sends=0; receives=0; routing_packets=0;
         highest_packet_id = 0;
         end_to_end_delay=0;

         for (i=0; i<MAX_PACKETS ; i++)
         { start_time[i] = 0; end_time[i]=0;}

         fin =  new FileInputStream ("final_sources"+h+".tr");
         DataInputStream br = new DataInputStream(fin);

         while ((thisLine = br.readLine()) != null) {
           // scan it line by line
           java.util.StringTokenizer st = new java.util.StringTokenizer(thisLine, " ");

           i=0;
           while(st.hasMoreElements())
           tokens[i++]= st.nextToken();

           if (tokens[0].equals("s") || tokens[0].equals("r")|| tokens[0].equals("f"))
              {
```

```
                // parse the time
                if (tokens[1].equals("-t")) time = Float.valueOf(tokens[2]).floatValue();

                // parse the packet_id
                if (tokens[39].equals("-Ii")) packet_id = Int-
nger.valueOf(tokens[40]).intValue();

                // calculate the sent packets
                if (to-
kens[0].equals("s")&&tokens[18].equals("AGT")&&tokens[34].equals("cbr")) sends++;

                // find the number of packets in the simulation
                if (packet_id >highest_packet_id) highest_packet_id = packet_id;

                // set the start time, only if its not already set
                if (start_time[packet_id] == 0) start_time[packet_id] = time;

                // calculate the receives and end-end delay
                if (tokens[0].equals("r") && tokens[18].equals("AGT") && to-
kens[34].equals("cbr"))
                      {
                         receives++;
                         end_time[packet_id] = time;
                      }

                      else end_time[packet_id] = -1;

                // calculate the routing packets
                if ((tokens[0].equals("s") || tokens[0].equals("f"))  && to-
kens[18].equals("RTR") &&  (tokens[34].equals("AODV") || tokens[34].equals("DSDV") || to-
kens[34].equals("OLSR") || tokens[34].equals("message") ))

                routing_packets++;

                }
             }
          // calculate the packet duration for all the packets
         for (packet_id = 0; packet_id <= highest_packet_id ; packet_id++) {

           packet_duration = end_time[packet_id] - start_time[packet_id];
           if (packet_duration >0) end_to_end_delay += packet_duration;
         }

        // calculate the average end-end packet delay
         avg_end_to_end_delay = end_to_end_delay / (receives );

         // calculate the packet delivery fraction
         pdfraction = ((float)receives/(float)(sends))*100;

         System.out.println(" \nsends "+sends);
         System.out.println(" receives "+receives);
         System.out.println(" routing overhead (packets) "+ routing_packets);
         System.out.println(" Normalized routing load
"+(float)routing_packets/(float)receives);
         System.out.println(" pdfraction " +pdfraction);
         System.out.println(" Avg End-End delay " +avg_end_to_end_delay);

         op.writeBytes(" " +sends);
         op.writeBytes(" "+receives);
         op.writeBytes(" "+ routing_packets);
         op.writeBytes(" "+(float)routing_packets/(float)receives);
         op.writeBytes(" " +pdfraction);
         op.writeBytes(" " +avg_end_to_end_delay);
         op.writeChar('\n');

         }
     }
       catch (Exception e) {
         e.printStackTrace();
         }
   }  }
```

LIST OF REFERENCES

1. J. Moy, "Open Shortest Path First (OSPF) Version 2", *IETF: The Internet Engineering Taskforce RFC 2328,* Apr 1998, last accessed on Jan 2006.

2. T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", *IETF: The Internet Engineering Taskforce RFC 3626,* Oct 2003, last accessed on Jan 2006.

3. The network simulator – ns-2.29, *http://www.isi.edu/nsnam/ns/*, last accessed on Sep 2005.

4. C. E. Perkins and E. Royer, "Ad-Hoc On-Demand Distance Vector Routing", *Proceeding of 2nd IEEE Workshop on Mobile Computing & Systems and Applications*, pp. 90–100, Feb 1999, last accessed on Sep 2005.

5. C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", *Computer Communications Review*, pp. 234–244, Oct 1994.

6. P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum and L. Viennot, "Optimized Link State Routing Protocol for Ad Hoc Networks", *IEEE INMIC*, pp. 62–68, Dec 2001, last accessed on Oct 2005.

7. K. Fall and K. Varadhan (Eds.), "The ns Manual", Jul 2005, *http://www.isi.edu/nsnam/ns/ns-documentation.html*, last accessed on Sep 2005.

8. F. J. Ros, "UM-OLSR Version 8.8.0", University of Murcia, Spain, 2006, *http://masimum.dif.um.es/?Software:UM-OLSR*, last accessed on Dec 2006.

9. "Distance Vector Algorithms", Connected: An Internet Encyclopedia, *http://www.freesoft.org/CIE/RFC/1058/6.html*, last accessed on Jan 2006.

10. C. Hedrick, "Routing Information Protocol (RIP)", *IETF: The Internet Engineering Taskforce RFC 1058,* Jun 1988, last accessed on Jan 2006.

11. Williams B. and Camp T., "Comparison of broadcasting techniques for mobile ad hoc networks", *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, pp.194–205, 2002.

12. Sung-Ju Lee, Mario Gerla, and Ching-Chuan Chiang, "On-Demand Multicast Routing Protocol", *IEEE Wireless Communications and Networking Conference (WCNC '99)*, Sep 21–24, 1999.

13. Y. Chun, L. Qin, L. Young and S. MeiLin, "Routing Protocols Overview and Design Issues for Self-Organized Network", *Proceedings of ICCT2000*, pp. 1298–1303, Aug 2000.

14. S.-J. Lee, *Routing and Multicasting Strategies in Wireless Mobile Ad hoc Networks,* Ph.D dissertation, University of California, Los Angeles, 2000.

15. S. J. Lee, W. Su, J. Hsu, M. Gerla and R. Bagrodia, "A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols", *Proceedings of IEEE INFOCOM 2000,* pp. 565–574, Mar 2000.

16. Daniel Lang, "A comprehensive overview about selected Ad Hoc Networking Routing Protocols", Department of Computer Science, Technische Universitat Munchen, Germany, Mar 2003, *www.cs.brown.edu/~ia/publications/2_conference/1999-12-MDA/survey.pdf*, last accessed on Nov 2005.

17. C.-C. Chiang, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel", *Proc. IEEE SICON '97*, pp. 197–211, Apr 1997.

18. D. Bein, A. K. Datta, C. R. Jagganagari and V. Villain, "A Self-stabilizing Link-Cluster Algorithm in Mobile Ad Hoc Networks", *Proceedings of The International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN),* pp. 436–441, Dec 2005.

19. E. Royer and C. Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", *IEEE Personal Communications*, Vol. 6, No. 2, pp. 46–55, Apr 1999.

20. Z.J. Haas, M.R. Pearlman and Prince Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks", IETF Internet draft, Jul 2002, *http://www.ietf.org/internet-drafts/draft-ietf-manet-zone-zrp-04.txt*, last accessed on Jan 2006.

21. M. Halvardsson and P. Lindberg, "Reliable group communication in a military Mobile Ad hoc Network", A Report from School of Mathematics and Systems Engineering, Vaxjo University, Sweden, Feb 2004, *www-ee.engr.ccny.cuny.edu/wwwb/mfecko/.../2004/hl04_rel_group_comm.pdf,* last accessed on Dec 2005.

22. Plesse, T., Lecomte, J., Adjih, C., Badel, M., Jacquet, P., Laouiti, A., Minet, P., Muhlethaler, P. and Plakoo, A., "OLSR performance measurement in a military mobile ad-hoc network", *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, pp. 704–709, Mar 2004.

23. Paul E. Black, "Bellman-Ford algorithm", from Dictionary of Algorithms and Data Structures, Paul E. Black, ed., NIST. *http://www.nist.gov/dads/HTML/bellmanford.html*, last accessed on Jan 2006.

24. C. E. Perkins and P. Bhagwat, "DSDV Routing over a Multihop Wireless Network of Mobile Computers", Chapter 3, pp. 53–74, in Charles E. Perkins, Ad Hoc Networking, Addison-Wesley, 2001.

25. Yi Lu, Weichao Wang, Yuhui Zhong, Bharat Bhargava, "Study of Distance Vector Routing Protocols for Mobile Ad Hoc Networks", *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom'03),* pp. 187–194, Mar 2003.

26. C. E. Perkins, Elizabeth M. Belding-Royer and I. D. Chakeres, "Ad hoc On-Demand Distance Vector (AODV) Routing", *http://www.ietf.org/internet-drafts/draft-perkins-manet-aodvbis-00.txt*, Oct 2003.

27. Zygmund J. Haas and Marc R. Pearlman, "The performance of query control schemes for the zone routing protocol", *IEEE/ACM Transactions on Networking*, Vol. 9, No. 4, pp. 427–438, Aug 2001.

28. Samir, Das, Robert, Castaneda, Jiangtao Yan, "Simulation-Based Performance Evaluation of Routing Protocols for Mobile Ad Hoc Networks", Mobile Networks and Applications, May 2000, pp. 179–189, *www.csee.umbc.edu/~krishna/cs628/Papers/p179-das.pdf*, last accessed on Jan 2006.

29. M. Gerla and J.T.-C. Tsai, "Multicluster, mobile, multimedia radio network", *ACM/Baltzer Journal of Wireless Networks. vol. 1*, (no. 3), pp. 255–265, 1995.

30. P. Krishna, N.H. Vaidya, M. Chatterjee and D.K. Pradhan, "A clusterbased approach for routing in dynamic networks", *ACM SIGCOMM Computer Communication Review*, pp. 49–65, Apr 1997.

31. F. Bai and A. Helmy, "A Survey of Mobility Models in Wireless Adhoc Network", University of Southern California, U.S.A, *nile.usc.edu/~helmy/important/Modified-Chapter1-5-30-04.pdf*, last accessed on Jan 2006.

32. J. Yoon, M. Liu and B. Noble, "Random Waypoint Considered Harmful", *in Proceedings of INFOCOM, IEEE*, Vol. 2, pp. 1312–1321, Apr 2003.

33. X. Hong, M. Gerla, G. Pei, and C.-C. Chiang, "A group mobility model for ad hoc wireless networks", *Proc. ACM Intern. Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM),* pp. 53–60, Aug 1999.

34. J. Kristoffersson, *Obstacle Constrained Group Mobility Model*, Master's thesis, Luleå University of Technology, Sweden, Dec 2005, *epubl.ltu.se/1402-1617/2005/306/index-en.html*, last accessed on Jan 2006.

35. Zygmunt J. Haas , Marc R. Pearlman, “The performance of query control schemes for the zone routing protocol”, *IEEE/ACM Transactions on Networking (TON)*, Vol. 9, Issue 4, pp. 427–438, 2001.

36. Christopher Dearlove, “OLSR Simulation, Implementation and Ad Hoc Sensor Network Application”, BAE SYSTEMS Advanced Technology Centre, U.K., 2004.

37. F. J. Ros, “MASIMUM - MANET Simulation and Implementation at the University of Murcia”, University of Murcia, *http://masimum.dif.um.es/?Software:UM-OLSR*, last accessed on Nov 2005.

38. Mounir Benzaid, Pascale Minety, Khaldoun Al Aghaz, “Integrating fast mobility in the OLSR routing protocol”, Report Number 4510, INRIA, Jun 2002, *http://www.inria.fr/rrrt/rr-4510.html*, last accessed on Feb 2006.

39. C. Gomez, D. Garcia, J. Paradells, “Improving Performance of a Real Ad-hoc Network by Tuning OLSR Parameters”, *Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC 2005)*, pp. 16–21, Jun 2005.

40. XGraph homepage, *http://www.isi.edu/nsnam/xgraph/*, last accessed on Oct 2005.

41. TraceGraph homepage, *http://www.geocities.com/tracegraph/*, last accessed on Oct 2005.

42. Additions to the *NS* network simulator to handle Ricean and Rayleigh fading, Department of Electrical and Computer Engineering at Carnegie Mellon University (CMU), *http://www.ece.cmu.edu/wireless/*, last accessed on Nov 2005.

43. R. J. Punnoose, P. V. Nikitin, and D. D. Stancil., “Efficient Simulation of Ricean Fading within a Packet Simulator”, *Vehicular Technology Conference*, Vol. 2, pp. 764–767, Sept 2000.

44. C. Waal and M. Gerharz, “BonnMotion: A Mobility Scenario Generation and Analysis Tool”, Communication Systems Group, Institute of Computer Science IV, University of Bonn, Germany, *http://web.informatik.unibonn.de/IV/Mitarbeiter/dewaal/BonnMotion/,* last accessed on Oct 2005.

45. K. Sadasivam, “Tutorial for Simulation-based Performance Analysis of MANET Routing Protocols in ns-2”, *nas.cl.uh.edu/yang/teaching/csci5931netSecuritySpr05/ns-tutorial.doc*, last accessed on Jan 2006.

46. I. Downard, "Simulating Sensor Networks in NS-2," NRL/FR/5522–04–10073, Naval Research Laboratory, Washington, D.C., May 2004.

47. A. McCabe, C. Dearlove and Leif Axelsson, “Scalability modelling of ad hoc routing protocols – a comparison of OLSR and DSR”, BAE Systems and Ericsson, 2004, *www.wireless.kth.se/adhoc04/proceedings/McCabe.pdf*, last accessed on Jan 2006.

48. V. Naumov and T. Gross, “Scalability of Routing Methods in Ad Hoc Networks”, Computer Systems Institute, ETH Zurich, Switzerland, 2005, *www.lst.inf.ethz.ch/research/publications/ publications/PERFORMANCE_2005/tech-report.html*, last accessed on Jan 2006.

49. G. Karbaschi and A. Fladenmuller, “Cross-layering for Performance Improvement in Multi-Hop Wireless Networks”, *Proceedings of the 8th International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN’05),* pp. 536–541, 2005.

50. X. Lin and N. B. Shroff, “The impact of imperfect scheduling on cross-layer rate control in wireless networks”, *Proceedings of IEEE INFOCOM 2005*, Vol. 3, pp. 1804–1814, Mar 2005.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Chairman, Code EC
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California

4. Professor John C. McEachen
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California

5. Professor Weilian Su
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California

6. Yap Lay Poh
   Defence Science & Technology Agency
   Singapore

7. Seah Peng Hwee
   Defence Science & Technology Agency
   Singapore

8. Pore Ghee Lye
   Defence Science & Technology Agency
   Singapore